

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE



In re REISSUE PATENT APPLICATION of

ATTWATER et al.

Atty. Ref.: 36-1008

Reissue of Patent No. 5,940,793

Granted: August 17, 1999

For: **VOICE-OPERATED SERVICES**

* * * * *

August 16, 2001

BOX REISSUE

Hon. Commissioner of Patents and Trademarks
Washington, DC 20231

Sir:

INFORMATION DISCLOSURE STATEMENT

The undersigned attorney brings to the attention of the Patent and Trademark Office the references listed on the attached form PTO-1449, a copy of each of which is enclosed. This is not to be construed as a representation that no better prior art exists, or that a reference is relevant merely because cited.

Some of the references were cited in various communications from foreign patent offices in respective counterpart foreign applications. An English language search report from one of the foreign patent offices citing some of the references is attached for the Examiner's information.


The Examiner's attention is also directed to the prior art in the parent application by the Applicant and/or Examiner for the reasons stated therein.

Reissue of Patent No. 5,940,793

The Examiner is requested to initial the attached form PTO-1449 and to return a copy of the initialed document to the undersigned as an indication that the attached references have been considered and made of record.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: 
Raymond Y. Mah
Reg. No. 41,426

RYM:sl
1100 North Glebe Road, 8th Floor
Arlington, VA 22201-4714
Telephone: (703) 816-4000
Facsimile: (703) 816-4100



P.B. 5818 - Patentaan 2
2280 HV Rijswijk (ZH)
☎ (070) 3 40 20 40
TX 31651 epo nl
FAX (070) 3 40 30 16

RECEIVED
02 NOV 1998
IP FORMALITIES
GROUP

Nederlandsches
Patentamt
Zweigstelle
in Den Haag
Recherchen-
Abteilung

European
Patent Office
Branch at
The Hague
Search
Division

ADS

Office européen
des brevets
Département A
La Haye
Division de la
recherche

J1017 U.S. PTO
09/930395
08/16/01

Lloyd, Barry George William
BT Group Legal Services,
Intellectual Property Department,
8th Floor, Holborn Centre,
120 Holborn
London EC1N 2TE
GRANDE BRETAGNE

DIARY REMINDER INFO
OTHER

Date/Date 22.10.98

Zeichen/Ref./Ref. A24909/EP	Anmeldung Nr./Application No./Demande n°/Patent Nr./Patent No./Brevet n° 95934749.3
Anmelder/Applicant/Demandeur/Patentinhaber/Proprietor/Titulaire BRITISH TELECOMMUNICATIONS public limited company	

COMMUNICATION

The European Patent Office herewith transmits

- ☐ the European search report
- ☐ the declaration under Rule 45 EPC
- ☐ the partial European search report under Rule 45 EPC
- ☒ the supplementary European search report concerning the international application under Article 157(2) EPC relating to the above-mentioned European patent application. Copies of the documents cited in the search report are enclosed.

The following specifications given by the applicant have been approved by the Search Division

- ☐ Abstract ☐ Title ☐ Figure
 - ☐ The abstract was modified by the Search Division and the definitive text is attached to this communication
 - ☐ The following figure will be published with the abstract, since the Search Division considers that it better characterises the invention than the one indicated by the applicant.
- Figure:
- ☐ Additional copy(copies) of the documents cited in the European search report.

REFUND OF THE SEARCH FEE

If applicable under Article 10 Rules relating to fees, a separate communication from the Receiving Section on the refund of the search fee will be sent later.



EPO Form 1507 02.93				



European Patent
Office

SUPPLEMENTARY
EUROPEAN SEARCH REPORT

Application Number
EP 95 93 4749

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl. 6)
X	EP 0 299 572 A (PHILIPS) 18 January 1989	1,2, 5-11, 20-22, 27-29	G10L5/06
Y	* page 2 - page 3, column 4, line 14 *	3	
Y,P	EP 0 625 758 A (IBM) 23 November 1994 * page 6, line 42 - page 7 *	3	
A	US 5 202 952 A (GILLICK ET AL.) 13 April 1993 * column 5, line 21 - column 6, line 10 *	4	
A	GB 2 165 969 A (BRITISH TELECOM) 23 April 1986 * page 1, line 33 - line 54 *	1,20-22, 27,28	
A	EP 0 269 233 A (SMITHS INDUSTRIES) 1 June 1988 * page 2 - page 3, column 4, line 4 *	1,20-22, 27,28	
A	YOUNG: "Use of dialogue, pragmatics and semantics to enhance speech recognition" EUROSPEECH 89, PARIS, FR, 26.-28. 9. 1989, vol. 9, no. 5-6, pages 551-564, XP002000047 ISSN 0167-6393, SPEECH COMMUNICATION, DEC. 1990, NETHERLANDS * paragraph 2.1 * * paragraph 2.2 *	1,20-22, 27,28	TECHNICAL FIELDS SEARCHED (Int.Cl. 6) G10L
X	CA 2 091 658 A (LENNIG ET AL.) 16 September 1994 * page 1 - page 6 *	12,14,17	
A	EP 0 484 070 A (IBM) 6 May 1992 * page 6, line 55 - line 58 *	12	
-/-			
The supplementary search report has been based on the last set of claims valid and available at the start of the search.			
Place of search THE HAGUE		Date of completion of the search 16 October 1998	Examiner Lange, J
CATEGORY OF CITED DOCUMENTS			
X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document		T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application C : document cited for other reasons A : member of the same patent family, corresponding document	



European Patent
Office

**SUPPLEMENTARY
EUROPEAN SEARCH REPORT**

Application Number
EP 95 93 4749

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int Cl 6)
A	EP 0 601 710 A (AMERICAN TELEPHONE & TELEGRAPH) 15 June 1994 * column 5 - column 6 *	12	
A	US 4 763 278 A (RAJASEKARAN ET AL.) 9 August 1988 * column 3, line 54 - column 4, line 11 *	15	
A	JP 06 204952 A (IBM) 22 July 1994 * the whole document * -& US 5 475 792 A (STANFORD ET AL.) 12 December 1995 * column 2, line 35 - column 3, line 12 *	18,19	
X	US 5 355 474 A (THURAISINGHAM ET AL.) 11 October 1994 * column 13, paragraph 2.2.2 *	26	
A	US 4 701 879 A (SCARR) 20 October 1987 * column 2, line 3 - line 31 *	26	
X	ANONYMOUS: "N Ary Join for Processing Query by Example. November 1976." IBM TECHNICAL DISCLOSURE BULLETIN, vol. 19, no. 6, November 1976, pages 2377-2381, XP002081147 New York, US * page 2378, paragraph 8 - page 2379, paragraph C *	26	TECHNICAL FIELDS SEARCHED (Int Cl 6)
X	EP 0 533 338 A (AMERICAN TELEPHONE & TELEGRAPH) 24 March 1993 * page 5, line 32 - line 43 *	33	
The supplementary search report has been based on the last set of claims valid and available at the start of the search.			

Place of search

THE HAGUE

Date of completion of the search

16 October 1998

Examiner

Lange, J

CATEGORY OF CITED DOCUMENTS

X: particularly relevant if taken alone
Y: particularly relevant if combined with another document of the same category
A: technological background
O: non-written disclosure
P: intermediate document

T: theory or principle underlying the invention
E: earlier patent document, but published on, or after the filing date
D: document cited in the application
L: document cited for other reasons
&: member of the same patent family, corresponding document



European Patent
Office

Application Number

EP 95 93 4749

CLAIMS INCURRING FEES

The present European patent application comprised at the time of filing more than ten claims.

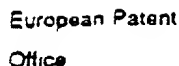
- ☐ Only part of the claims have been paid within the prescribed time limit. The present European search report has been drawn up for the first ten claims and for those claims for which claims fees have been paid, namely claim(s):
- ☐ No claims fees have been paid within the prescribed time limit. The present European search report has been drawn up for the first ten claims.

LACK OF UNITY OF INVENTION

The Search Division considers that the present European patent application does not comply with the requirements of unity of invention and relates to several inventions or groups of inventions, namely:

see sheet 8

- ☒ All further search fees have been paid within the fixed time limit. The present European search report has been drawn up for all claims.
- ☐ Only part of the further search fees have been paid within the fixed time limit. The present European search report has been drawn up for those parts of the European patent application which relate to the inventions in respect of which search fees have been paid, namely claims:
- ☐ None of the further search fees have been paid within the fixed time limit. The present European search report has been drawn up for those parts of the European patent application which relate to the invention first mentioned in the claims, namely claims:



The Search Division considers that the present European patent application does not comply with the requirement of unity of invention and relates to several inventions or groups of inventions, namely:

- 1. claims 1-11, 20-25, 27-32 :**

All the technical features from claims 1-2 are known from D1. So the technical problem in the light of this document is to construct a speech recogniser taking into account the number of syntactic connections between words (claim 3). This is solved by employing lists of connected words to restrict the search space and so providing additional syntactic information.

- 2. claims 12-19 :**

deal with the problem to construct a speech recogniser resp. a speaker recogniser/verifier (cl. 15) using additional information to enhance the recognition rate. This is solved by using information about the origin of a telephone call.

- 3. claim 26**

deals with the problem to provide a method of data retrieval using additional information to enhance the retrieval rate. This is solved by employing lists of connected data to restrict the search space and so providing additional syntactic information.

4. claim 33.

The problem is to provide a method of speech recognition dealing with unreliable recognition results. The solution is to fall back to stored speech data and employing another recognition process therewith.

[illegible]

ANNEX TO THE EUROPEAN SEARCH REPORT
ON EUROPEAN PATENT APPLICATION NO.

EP 95 93 4749

PAGE 10

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information

16-10-1998

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0299572 A	18-01-1989	DE 3723078 A	19-01-1989
		DE 3886111 D	20-01-1994
		JP 1078299 A	23-03-1989
		US 4947438 A	07-08-1990
EP 0625758 A	23-11-1994	GB 2277387 A	26-10-1994
		JP 7093372 A	07-04-1995
US 5202952 A	13-04-1993	AT 158889 T	15-10-1997
		CA 2085895 A	23-12-1991
		DE 69127818 D	06-11-1997
		DE 69127818 T	30-04-1998
		EP 0535146 A	07-04-1993
		JP 6501319 T	10-02-1994
		US 5526463 A	11-06-1996
		WO 9200585 A	09-01-1992
GB 2165969 A	23-04-1986	NONE	
EP 0269233 A	01-06-1988	GB 2198572 A	15-06-1988
CA 2091658 A	16-09-1994	CA 2119072 A	16-09-1994
		US 5479488 A	26-12-1995
EP 0484070 A	06-05-1992	JP 4248598 A	04-09-1992
		US 5367609 A	22-11-1994
EP 0601710 A	15-06-1994	US 5392343 A	21-02-1995
		JP 6225024 A	12-08-1994
US 4763278 A	09-08-1988	EP 0125422 A	21-11-1984
JP 6204952 A	22-07-1994	JP 2524472 B	14-08-1996
		US 5475792 A	12-12-1995
US 5355474 A	11-10-1994	US 5481700 A	02-01-1996
		US 5694590 A	02-12-1997
US 4701879 A	20-10-1987	GB 2161263 A	08-01-1986

EP 0 95 93 4749

For more details about this annex : see Official Journal of the European Patent Office, No 12/82

CP 95 93 4749

16-10-1998

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0533338 A	24-03-1993	CA 2069599 A,C JP 5249993 A	17-02-1993 28-09-1993

[54] **SPEAKER-INDEPENDENT WORD RECOGNIZER**

[75] Inventors: Periagaram K. Rajasekaran; George R. Doddington, both of Richardson, Tex.

[73] Assignee: Texas Instruments Incorporated, Dallas, Tex.

[21] Appl. No.: 484,820

[22] Filed: Apr. 13, 1983

[51] Int. Cl.⁴ G10L 5/00

[52] U.S. Cl. 364/513.5; 381/43

[58] Field of Search 381/41, 42, 43, 44, 381/45, 46, 47, 48, 49, 50; 364/513, 513.5

[56] **References Cited****U.S. PATENT DOCUMENTS**

3,553,372	1/1971	Wright et al.	381/43
4,256,924	3/1981	Sakoe	381/43
4,363,102	12/1982	Holmgren et al.	364/513
4,477,925	10/1984	Avery et al.	381/43
4,528,688	7/1985	Ichikawa et al.	381/43

OTHER PUBLICATIONS

Flanagan, Speech Analysis, Synthesis and Perception Springer-Verlag, New York, 1972, pp. 192-204.

Primary Examiner—Emanuel S. Kemeny
Attorney, Agent, or Firm—William E. Hiller; N. Rhys Merrett; Melvin Sharp

[57] **ABSTRACT**

Speaker-independent word recognition is performed, based on a small acoustically distinct vocabulary, with minimal hardware requirements. After a simple preconditioning filter, the zero crossing intervals of the input speech are measured and sorted by duration, to provide a rough measure of the frequency distribution within each input frame. The distribution of zero crossing intervals is transformed into a binary feature vector, which is compared with each reference template using a modified Hamming distance measure. A dynamic time warping algorithm is used to permit recognition of various speaker rate, and to economize on the reference template storage requirements. A mask vector for each reference template is used to ignore insignificant (or speaker-dependent) features of the words detected.

18 Claims, 5 Drawing Sheets0 **DETERMINE THE FEATURE VECTOR AT THE REFERENCE TIMES FOR EACH SAMPLE**

0	1	1	0	0	1	0	1	--	SAMPLE 1 FV
1	0	1	0	1	1	0	1	--	SAMPLE 100 FV

ADD COLUMNWISE

3	57	96	97	41	98	95	33	--	SUM VECTOR
---	----	----	----	----	----	----	----	----	------------

(BEHAVIOR DISTRIBUTION)

0 **CHOOSE AN "UNANIMITY FACTOR" -- E.G. 93%**

0	X	1	1	X	1	1	X	--	RF(1) REFERENCE VECTOR
1	0	1	1	0	1	1	0	--	RN(1) MASK VECTOR

(SIGNIFICANT GROUP BEHAVIOR)

J1017 U.S. PTO
 09/930395
 08/16/01



RECEIVED
 SEP 14 1988
 U.S. PATENT & TRADEMARK OFFICE

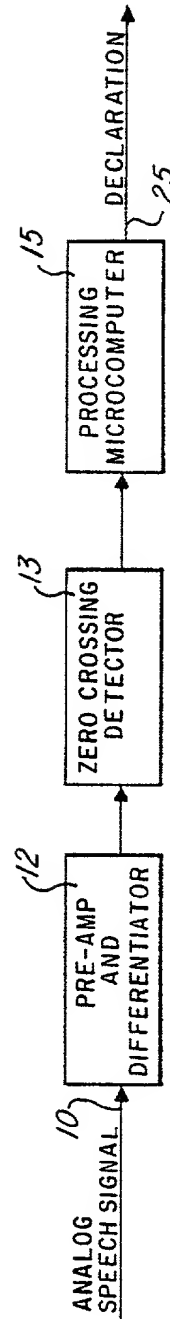
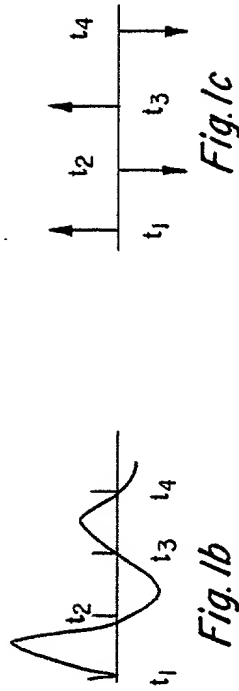


Fig. 1a

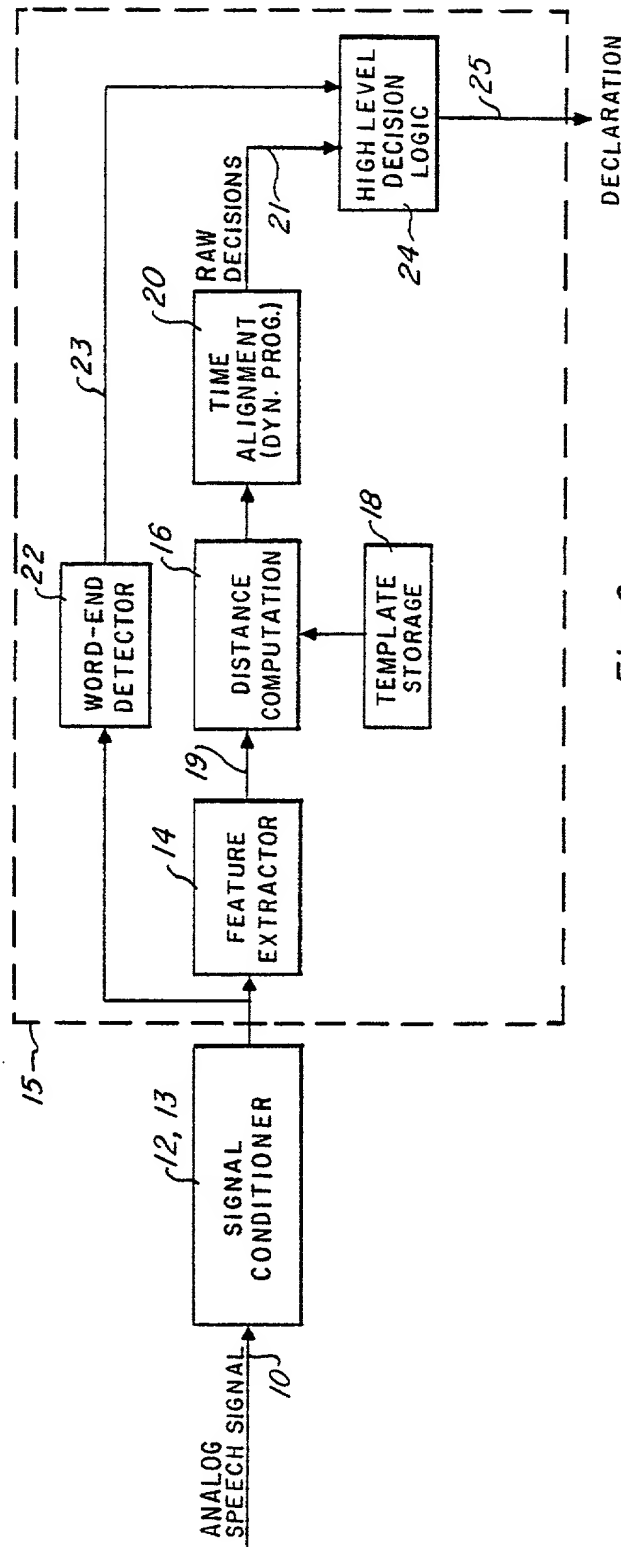


Fig. 2

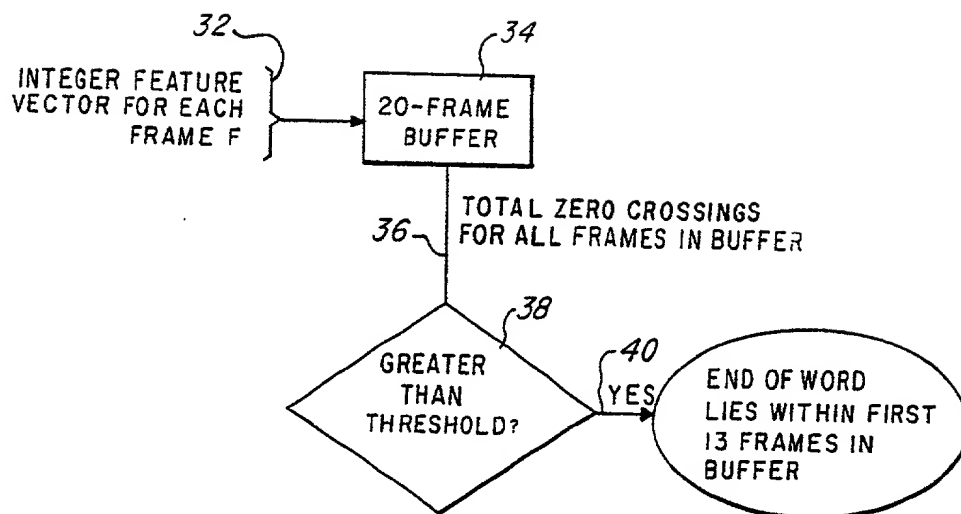


Fig. 3

0 DETERMINE THE FEATURE VECTOR AT THE REFERENCE TIMES FOR EACH SAMPLE

0	1	1	0	0	1	0	1	--	SAMPLE 1 FV
:									
1	0	1	0	1	1	0	1	--	SAMPLE 100 FV

ADD COLUMNWISE

3	57	96	97	41	98	95	33	--	SUM VECTOR
---	----	----	----	----	----	----	----	----	------------

(BEHAVIOR DISTRIBUTION)

0 CHOOSE AN "UNANIMITY FACTOR" -- E.G. 93%

0	X	1	1	X	1	1	X	--	RF(1) REFERENCE VECTOR
1	0	1	1	0	1	1	0	--	RM(1) MASK VECTOR

(SIGNIFICANT GROUP BEHAVIOR)

Fig. 4

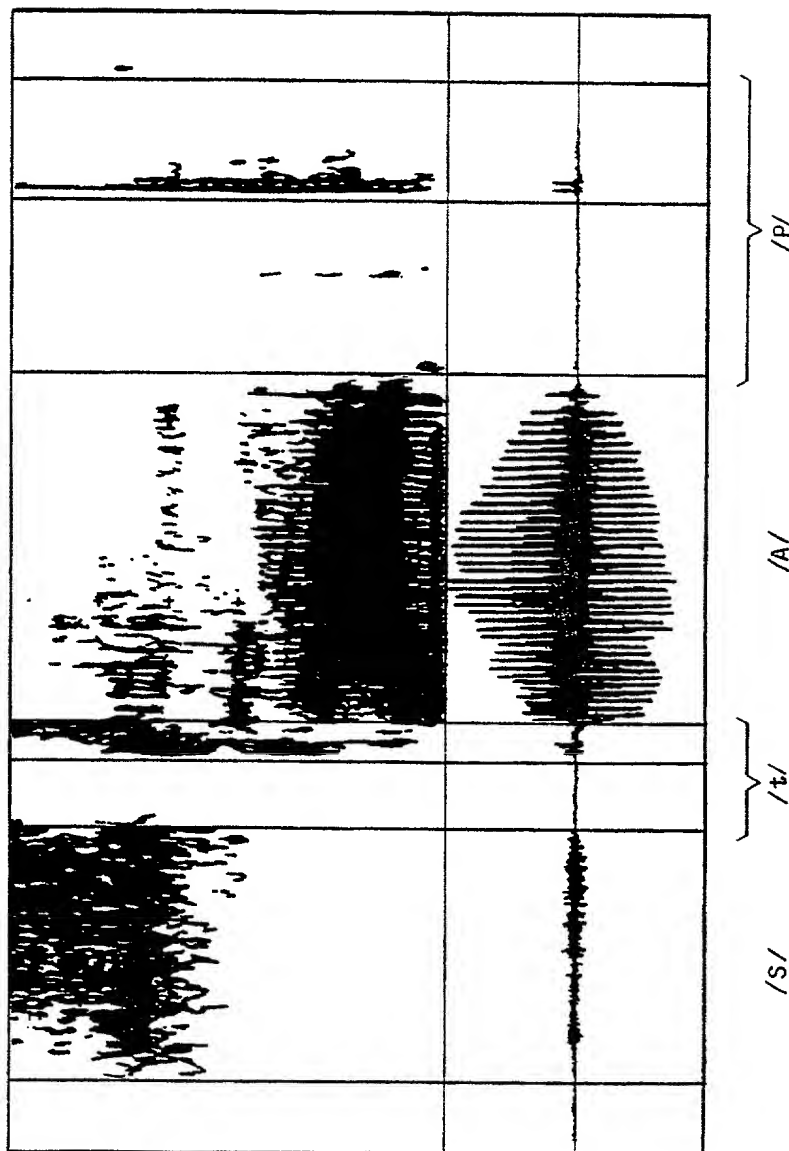


Fig. 5

SECRET

SPEAKER-INDEPENDENT WORD RECOGNIZER

BACKGROUND AND SUMMARY OF THE INVENTION

The present invention relates to a speaker-independent speech recognizer, that is to a machine capable of automatically recognizing and decoding speech from an unknown human speaker.

There are many applications where it would be highly desirable to have such a speaker-independent speech recognizer configured for a small vocabulary. For example, such a word recognizer would be extremely useful for automotive controls and video games. If even a very small control vocabulary were available, many non-critical automotive control functions which frequently require the driver to remove his eyes from the road could be done by direct voice inputs. Control of a car's radio or sound system could be usefully accomplished in this matter. The more sophisticated monitoring and computational functions available in some cars could also be more efficiently met with a voice query/voice output system. For example, if a driver could say "fuel", and have his dashboard reply verbally "seven gallons—refuel within 160 miles," this would be very convenient in automotive control design. Similarly, an arcade video game could be designed to accept a limited set of verbal inputs such as "shoot", "pull up", "dive", "left", and "right". These applications, like many others, are extremely cost sensitive.

Thus, to provide a word recognizer for the large body of applications of this type, it is not necessary that the recognizer be able to recognize a very large vocabulary. A small vocabulary, e.g. 6 to 20 words, can be extremely useful for many applications. Secondly, it is not necessary that a word recognizer for such applications be able to recognize a word embedded in connected speech. Recognition of isolated words is quite sufficient for many simple command applications. Third, in many such applications, substitution errors are much more undesirable than rejection errors. For example, if a consumer is making purchases from a voice-selected vending machine, it is much more desirable to have the machine reply "input not understood" than to have the machine issue the wrong item.

Thus, it is an object of the present invention to provide a low cost word recognizer system which has a very low rate of substitution errors.

It is highly desirable to have such word recognizer systems operate with a low computational load. In many attractive applications, a modest error rate can easily be tolerated (e.g. 85% accurate recognition), but the cost requirements are stringent. Thus, it would be highly desirable to have a word recognizer which could be implemented with an ordinary cheap 8 bit microcomputer, together with cheap analog chips, but without requiring any high speed chips or dedicated processors. Of course, it is always possible to do speaker-independent word recognition using a minicomputer or a main frame, but such an implementation has no practical relevance to most of the desirable applications, since most of the applications are cost-sensitive.

Thus, it is an object of the present invention to provide a speaker-independent word recognizer which can be implemented with an ordinary 8-bit microcomputer, and does not require any high-speed or special-function processing chips.

It is a further object of the present invention to provide a speaker-independent word recognizer for a limited vocabulary which can be implemented using an 8-bit microcomputer and analog chips.

A further problem in speaker-independent recognition has been the preparation of an appropriate set of templates. Any one speaker, or any set of speakers with a common regional accent, may pronounce a certain word consistently with certain features which will not be replicated in the general population. That is, the reference templates for speaker-independent vocabulary must not specify any feature of a word which is not a strictly necessary feature. It is always possible to prepare a set of reference templates using empirical optimization, but this can be immensely time consuming, and also precludes the possibility of user-generation of reference templates in the field.

Thus, it is a further object of the present invention to provide a speech recognizer, for which the preparation of reference templates requires minimal empirical input from trained researchers.

It is a further object of the present invention to provide a method for preparing vocabulary templates for a speaker-independent word recognizer which can be implemented by minimally skilled users.

A further difficulty in preparing a speaker-independent word recognizer for cost-sensitive systems is memory requirements. That is, it is highly desirable in many systems where small microcomputers are to be used not to tie up too much program memory with the word recognition algorithm and templates. In particular, in many applications for portable devices (e.g. a calculator or watch which can receive spoken commands), the power requirements of unswitched memory impose a critical constraint. Since speech vocabulary templates must be saved during power-off periods, the amount of memory (CMOS or nonvolatile) required for speech reference templates is a very important cost parameter.

Thus, it is a further object of the present invention to provide a speaker-independent word recognizer which has absolutely minimal memory requirements for storing reference templates.

A further problem in any word recognizer, which is most particularly important in a speaker-independent word recognizer, is that speakers will typically vary, not only in their average rate of speech, but in their timing of the syllable within a given word. Since this information is not normally used by human listeners in making a word recognition decision, it will typically vary substantially among the speech patterns of different speakers. It is therefore necessary that a speaker-independent word recognizer be insensitive to a reasonable range of variation in the average rate and localized timing of human speech.

It is therefore a further object of the present invention to provide a speaker-independent word recognizer which is reasonably insensitive both to average rate and to localized variations in timing of human speech.

It is a further object of the present invention to provide a speech recognition system which is reasonably insensitive both to average rate and to localized variations in timing of human speech, which can be implemented using a simple microcomputer with no expensive custom parts required.

A further characteristic which it would be desirable to implement in a speaker-independent word recognition system is the capability for vocabulary change. Thus, for example, in a calculator which can be ad-

dressed by spoken commands, it would be desirable to have the set of spoken commands be variable with different modules (for example), or to be user variable as user-customized software is loaded into the calculator.

However, to accomplish this, it is desirable that the reference template set preparation be based on reasonably simple exclusion algorithms, so that a reasonably unskilled user can prepare a new template set. It is also necessary that the template set be addressable, so that templates can be downloaded and substituted.

It should also be noted that the capability to change templates is sensitive to the memory space required for each template. That is, if the memory templates can be stored reasonably compactly, then a mask location can be used to indicate which subset of all possible stored templates corresponds to the currently active vocabulary. Thus, for example, in an automotive control system, a master vocabulary might contain only a set of words indicating various areas of control functions, such as "radio", "wipers", "engine", "computer", etc. After any one of these function areas have been selected, a new localized set of reference templates would then be used for each particular function area. Each localized set of reference templates would have to include one command to return to the master template set, but otherwise could be fully customized. Thus, a localized set of commands for radio control could include such commands as "FM", "AM", "higher", "lower", "frequency", "volume", etc.

Thus, it is a further object of the present invention to provide a speaker-independent recognizer which functions on a limited vocabulary, but in which the vocabulary set can be easily changed.

It is a further object of the present invention to provide a speaker-independent recognizer which functions on a limited vocabulary, but in which the vocabulary set can be easily changed, which can be implemented using simple commercially available microcomputer parts.

A further desirable option in speaker-independent word recognizer systems is the capability to function in a speaker dependent mode. That is, in such applications as automobile controls or speech-controlled calculators, it is necessary that the systems be shipped from the factory with a capability to immediately receive speech input. However, many such devices will typically be used only by a limited set of users. Thus, it is desirable to be able to adapt the template set of speaker-independent device to be optimized for a particular user or group of users. Such re-optimization could be used to increase the vocabulary size or lower the error rate in service, but requires that the process of modifying templates be reasonably simple.

Thus, it is a further object of the present invention to provide a speaker-independent word recognizer which can be re-optimized easily to operate in a speaker dependent mode, for a specific speaker or for a limited group of speakers.

Thus, it is a further object of the present invention to provide a speaker-independent word recognizer, which can be easily re-optimized to operate in a speaker dependent mode for a specific speaker or for a limited group of speakers, and which can be economically configured using a simple microcomputer and simple analog parts.

Speaker-independent word recognition is performed, based on a small acoustically distinct vocabulary, with minimal hardware requirements. After a simple preconditioning filter, the zero crossing intervals of the input

speech are measured and sorted by duration, to provide a rough measure of the frequency distribution within each input frame. The distribution of zero crossing intervals is transformed into a binary feature vector, which is compared with each reference template using a modified Hamming distance measure. A dynamic time warping algorithm is used to permit recognition of various speaker rates, and to economize on the reference template storage requirements. A mask vector for each reference template is used to ignore insignificant (speaker-dependent) features of the words detected.

To achieve these and other objects of the invention, the present invention comprises:

A word recognizer, comprising:

input means for receiving an analog input signal corresponding to speech;

a signal processor, said processor conditioning said input signal according to a predetermined characteristic and measuring zero crossing intervals of said conditioned signal to provide a binary feature vector;

distance measurement means, said distance measurement means comparing said binary feature vector with each of a plurality of binary reference vectors (said reference vectors being organized in sequences corresponding to words) to provide a distance measure at least partially corresponding to a Hamming distance measure with one of said feature vectors; and

recognition means for recognizing words in accordance with the sequence of said distance measures between each said sequence of said reference vectors and successively received ones of said feature vectors.

According to a further embodiment of the present invention, the present invention comprises:

A word recognizer, comprising:

input means for receiving an analog input signal corresponding to speech;

a signal processor, said processor conditioning said input signal according to a predetermined characteristic and measuring zero crossing intervals of said conditioned signal to provide a feature vector;

distance measurement means, said distance measurement means comparing said feature vector with each of a plurality of reference vectors (said reference vectors being organized in sequences corresponding to words) to provide a distance measure at least partially corresponding to a Hamming distance measure with respect to said reference vectors for each successive one of said feature vectors;

recognition means for recognizing words in accordance with the sequence of said distance measures between each said sequence of said reference vectors and successively received ones of said reference vectors and successively received ones of said feature vectors, said recognizer also performing a dynamic programming step to provide an optimal subsequence match between successively received ones of said feature vectors and said sequences of reference vectors.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be described with reference to the accompanying drawings, wherein:

FIG. 1a shows a block diagram of the word recognizer of the present invention;

FIG. 1b is a graphical representation of an analog speech signal with respect to time after initial conditioning thereof, but prior to zero-crossing detection, in the word recognizer of FIG. 1a;

FIG. 1c is a schematic representation of the speech signal of FIG. 1b, following its subjection to zero-crossing detection, in the word recognizer of FIG. 1a;

FIG. 2 shows a block diagram of the preferred hardware implementation of the word recognizer according to the present invention;

FIG. 3 is a schematic diagram indicative of the end of word window operation used to identify word endings in the preferred embodiment of the present invention;

FIG. 4 shows a schematic indication of the processing of raw speaker inputs to achieve the mask vector for a reference template set, according to an empirical unanimity factor; and

FIG. 5 shows an example of the classification of a speech input according to its acoustic segmentation.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention includes several points of novelty, and also can be implemented in numerous different ways. Thus, the following description will suggest a number of modifications and variations of the present invention, without thereby implying that the present invention is limited to any specific embodiments thereof.

FIG. 1a shows generally the organization of the operations used in the word recognizer of the present invention. That is, a raw speech waveform 10 in the form of an analog speech signal is first subjected to signal conditioning including extremely simple prefiltering operations, e.g. to reject out of band signals, and wherein a pre-amplifier and an analog differentiator 12 may further act upon the analog speech signal 10. The speech signal may then generally take the form of the waveform illustrated in FIG. 1b. In the latter respect, it will be observed from FIG. 1b that the analog signal is generally sinusoidal and traces an undulating path above and below a "zero" polarity axis during the entire time duration including the respective time instants t_1 , t_2 , t_3 , and t_4 . The time instants t_2 , t_3 , and t_4 identify the zero-crossings of the waveform illustrated in FIG. 1b in which a transition occurs in the polarity sign of the waveform. Thus, the time instant t_2 identifies the zero-crossing of the waveform as it moves from a "plus" polarity to a "minus" polarity; t_3 identifies the zero-crossing of the waveform as it moves from a "minus" polarity to a "plus" polarity; and t_4 identifies the zero-crossing of the waveform as it moves from a "plus" polarity to a "minus" polarity.

Signal conditioning of the analog speech signal continues by monitoring the pre-amplified and differentiated speech signal with a zero-crossing detector 13 to sense each zero-crossing of the speech signal. The zero-crossing detector 13 duly counts each polarity transition in the speech signal and assigns a time instant when each such zero-crossing occurs. FIG. 1c schematically represents the speech signal of FIG. 1b, by indicating the signal polarity at each of the time instants t_1 , t_2 , t_3 , and t_4 , as sensed by the zero-crossing detector 13.

After the signal conditioning effected by the pre-amplifier and analog differentiator 12, and the zero-crossing detector 13, the conditioned speech signal is exposed to a signal processing unit 15, which may take the form of a microcomputer, for further signal process-

ing to enable electronic word recognition of the analog input speech signal 10 to take place. Word recognition is generally indicated by the declaration output 25 from the microcomputer 15 as determined by the decision logic thereof. In the latter respect, referring to FIG. 2, the signal processing unit 15 is illustrated with dashed lines and includes a feature extractor 14 which receives the conditioned speech signal as an input. The feature extractor 14, in the presently preferred embodiment, simply measures the intervals between zero crossings of the digital wave form received from the signal conditioner 12, 13, and then simply sorts the various zero crossing interval measurements received during any one frame into bins, to provide an integer feature vector which gives a rough measurement of frequency distribution during that frame. The elements of the integer feature vector are then compared with various thresholds, to provide a binary feature vector. This provides the basic feature measurement. Note that no digital-to-analog conversion is required.

The distance measurement 16 then compares the feature vector provided by the feature extractor 14 as an output along the line 19 with the feature vector received from a template storage portion 18. It is important to note two key features of the invention at this time. First, the storage 18 contains not only a feature vector for each template but also a mask vector. The mask vector is used to introduce don't care weightings into the feature vector stored with each reference template, as will be discussed below. Thus, the set of features of an input frame on which comparison for recognition is performed is selected, and can vary for each frame of each word template. Note that the various word templates stored in storage 18 each comprise a sequence of frames. That is, in a typical case each word template might comprise a sequence of 8-12 frames. (Each of the reference frames is expected to correspond to 2 of the 20 millisecond input frames, but can be warped to correspond to only one of the frames or to as many as 4 of the input frames, as described below.) The time alignment operation 20 selects the best match of each of the reference templates to the current input frame sequence, and provides running measurements of the raw match as output. a word end detector 22 also provides along line 23 the input to high level decision logic 24, and the word end measurement, together with the running cumulative word fit provided by the time alignment block 20, provide the basis for the high level decision logic 24 to make the word recognition decision which is provided as a declaration output 25.

The operation of these various components of the invention will now be discussed in greater detail. The speech signal 10, which is typically raw analog input from a microphone (and typically a preamplifier) is provided to a signal conditioner 12, 13.

The filter functions preferably performed by the signal conditioner 12, 13 include only an extremely simple filtering operation. In the presently preferred embodiment, the signal conditioner 12, 13 comprises a low pass filter with a corner frequency of 6.25 KHz to reject out of band signals an analog differentiator, and a Schmitt trigger. The differentiator effectively emphasizes the high frequency components in the input signal. That is, the zero crossing characteristics of a signal can easily be dominated by a strong low frequency component, and the use of the first derivative as the function on which zero crossing analysis is performed minimizes this problem.

It should be noted that the filtering functions are not necessarily so minimal. In particular, the zero crossing characteristics of a speech signal are highly sensitive to the frequency preemphasis and also to the phase shifting introduced by a prefiltering section, and a wide variety of prefiltering characteristics may optionally be used, e.g., to provide a more critical distinction between the words in a given vocabulary set or to reject particular noise characteristics. That is, the prefiltering characteristics will substantially affect the extent to which perceptually distinct input frames are measurably distinct in the very limited information provided by the recognition algorithm of the present invention. A wide variety of such filtering characteristics could be introduced in modifications and variations of the present invention, but the principal embodiment of the present invention uses only simple processing functions as noted.

In addition, bandpass filtering can also be used in the signal conditioner, to reject out of band signals, although this is not used in the presently preferred embodiment.

It should be noted that the Schmitt trigger performs a rather important signal processing function, namely center-clipping. That is, where zero crossings to a function including noise are being measured, even a very low noise power, at moments when the function value is near zero, can introduce many spurious zero crossings. To avoid this problem in recognition, center-clipping (using the hysteresis characteristics of the Schmitt trigger) in effect ignores zero crossings unless the waveform reaches a certain minimum value between two adjacent zero crossings. Although a Schmitt trigger is not the only way to accomplish this center-clipping, some such function in the signal conditioner is highly desirable, since it greatly reduces the noise in the low-interval zero crossings.

The actual zero crossing information can be obtained in a variety of ways, as is obvious to those skilled in the art. For example, the analog input signal can be applied to the Schmitt trigger mentioned, or to a polarity sensing saturated output amplifier, to provide a strongly clipped signal, i.e., a sequence of rectangular waveforms of alternating sign. These waveforms can then be converted to logic levels and provided as inputs to a microcomputer which counts the duration of each rectangular waveform portion (that is, the duration of each interval between zero crossings) in terms of clock cycles of the microcomputer. Of course, this function could easily be embodied in SSI logic, with a flip-flop and counters, or otherwise, but the embodiment in a microprocessor or microcomputer is preferred. The clock resolution of the microprocessor is preferably plus or minus 40 microseconds or less, but most commercial microprocessors can meet this. For example, an 8080, a Z-80, or a TMS 7000 would all be suitable.

The next step in processing the speech signal is to generate counts of the zero-crossing interval distribution in each frame of a sequence of frames, spaced at a frame period. In the presently preferred embodiment, the frame period is 20 msec, but this frame period can easily be varied. If a longer frame period is used, rapid speech may not be well recognized, but this may be an acceptable tradeoff in some applications for the sake of lower processor load. Similarly, a shorter frame period imposes a higher processor load, but provides a relatively slight gain in performance. Thus, frame periods in the range of 1 to 200 msec are within the scope of the present invention.

It should be noted that the input is not necessarily even divided into frames prior to this stage. That is, an advantage of using a microprocessor to measure the zero crossing intervals is that the microprocessor can at the same time impose the initial division of the analog input signal into frames.

At each frame, a feature vector is generated from the input as follows: first, the RMS energy of the analog signal is measured over an interval which need not exactly coincide with the frame period. For example, in the presently preferred embodiment, the energy is measured over an analysis window of 30 msec. This provides some smoothing of the energy values between frames, and precludes missing short high-energy events. In addition, the zero crossing intervals are classified at this time. Again, the analysis window over which the characteristics of the zero crossings are measured need not be exactly the same as the frame period, and is 30 msec in the presently preferred embodiment.

Thus, to generate the feature vector, the zero crossing intervals of the analog waveform over a 30 msec interval are examined. The presently preferred method of extracting a feature vector from the multiple zero crossing interval numbers is as follows, but of course a wide range of other expedients could be used to provide a feature vector representative of the distribution of zero crossing intervals. In the presently preferred embodiment, the zero crossing intervals within the 30 msec analysis waveform are sorted into four "bins", where each bin generally corresponds to a bandpass filter. That is, bin 1 counts the number of zero crossing intervals within the analysis window which have durations between seven and 13 samples (one sample is equal to 80 microseconds in this embodiment); bin 2 counts the number of zero crossing intervals in the analysis window with a duration between four and six intervals; bin 3 counts the number of zero crossing intervals in the analysis window with a duration of two or three samples; and bin 4 counts the number of zero crossing intervals in the analysis window which have a duration of one sample. These numbers are preferably accumulated by the microcomputer as the clipped rectangular waveform is received, so that the actual durations of the various zero crossings need not be stored at any point. That is, when the clipped input waveform changes sign, the microcomputer preferably notes the number of clock pulses since the last change of sign, increments the count in the appropriate bin by one, and resets its count register and begins to count the number of clock pulses until the next zero crossing. Thus, the number of zero crossings counted in any one "bin" corresponds generally to the energy which would have been measured through a corresponding bandpass filter, and the distribution over all of the bins provides an analog feature vector, which in the presently preferred embodiment includes four analog numbers.

Next, this integer feature vector is converted to a binary feature vector as follows. For example, the count found in bin 3 is compared to two thresholds to generate elements five and six of the binary feature vector: if the count is greater than a threshold B3L, then element 5 of the binary feature vector is set at 1 (and otherwise remains at zero); if the count in bin 3 is less than a second threshold B3U, then a 1 is entered in element 6 of the binary feature vector, which also otherwise remains at zero). That is, each bin has lower and upper thresholds, which are empirically chosen to maximize the discrimination between words used. In the presently preferred

embodiment, the eight thresholds used, expressed in number of sample values, are:

Bin number	Lower threshold	Upper Threshold
1	1	4
2	4	8
3	8	16
4	16	32

Again, it should be noted that the presently operational embodiment has used very frequent high density resolution sampling as an initial step, and hence the zero crossing intervals are expressed in samples, but the contemplated best mode of the present invention would not use such expensive high-rate high-resolution sampling, and would use analog stages initially instead, as discussed above.

Thus, the foregoing process has produced a feature vector (eight bits in the presently preferred embodiment) for each frame of the input signal. This feature vector is compared with various reference vectors according to a distance measure, and word recognition is made in accordance with the sequence of distance measures between the sequence of reference vectors in a word template and all or some of the sequence of input feature vectors received.

The distance measure used is essentially a Hamming distance measure between the input frame and any particular reference frame, but there is one important additional point of novelty in the distance measure. A mask vector is preferably stored along with each reference vector, to mask the less significant elements of the reference vectors. Thus, the actual template for a word consists of a sequence of pairs of binary vectors: for each reference frame in the word template, both a reference feature vector and a mask vector are stored. For example, if the fourth element of a mask vector is 1, then the fourth element of the corresponding reference vector is used in the distance computation. If the mask vector element is zero, the corresponding element of the associated reference vector is not used in the distance computation. Thus, the distance between the test feature vector TF, and the *i*-th reference vector RF(*i*) and mask vector RM(*i*) is defined by the following logical operation:

$$DTF_i = \text{Hamming}(\text{RM}(i), \text{and} . (\text{TF.xor.RF}(i)))$$

Thus, DTF_i is the Hamming distance between the test vector TF and the *i* vector set of the template for the given word. This distance indicates the number of similarities between test feature vector TF and reference vector RF(*i*), with masking defined by the zero valued elements in the mask vector RM(*i*).

It should be recognized that this use of a mask vector to exclude insignificant components of each reference vector is broadly novel, and may be modified and varied widely. For example, it is not strictly necessary that the feature and reference vectors be binary, since a binary masking value may be used to mask the results of an analog subtraction step as well. In fact, it is not even strictly necessary that the mask vector itself be binary, although this is greatly preferable. If the mask vector is allowed to take on analog values, then it functions essentially as a weight vector. A weighting vector is still useful for disregarding insignificant bits in a recognition comparison, but an analog weighting vector does not offer nearly the computational efficiency which is provided by a binary mask vector. Moreover, preparations

of a binary mask vector for a given word recognition set can be performed very simply and efficiently, as will be described below.

In addition, it should be noted that the novelty in the use of a masking vector is not by any means limited to use of an eight-bit feature vector, nor to recognition applications where the essential feature vector extraction step is based on zero crossing intervals, but can be applied to any speech recognition system whatsoever.

The method by which the reference vectors in a word template are generated will now be described.

To construct a template, the starting point is a large number of independent samples of that word as pronounced by a population which is sufficiently diverse to approximate that whose speech the recognizer will be required to recognize. For example, if the speech recognizer in use will be exposed to spoken inputs from men, women and children having all of the common regional accents, then the initial data base should also be obtained from men, women, and children, and should also include as many as possible of the regional accent types which must be recognized in use.

Correspondingly, if the recognizer is to be operated in a speaker-dependent mode or is to be customized for a small group of speakers, the number of samples must remain large, but the speakers within the relevant set will be proportionately represented. For example, if four speakers are to be recognized, each should contribute an average of 25 samples to the data base.

The first step is a manual classification step. Suppose that a template is to be constructed for the word "stop". This word has six distinct acoustic segments as shown in the spectrogram of FIG. 5. These are the initial sibilant/s/, stop and release/t/, vowel portion/A/, and the final stop and release/p/. These six segments are preferably marked interactively, with the aid of spectrogram and waveform plots, on graphics terminals, for each sample in the data base. Thus, this step manually establishes the location of corresponding acoustic segments within the data base sampled. This step is used because various speakers will vary the relative length of different acoustic segments within a word, and it is necessary, when estimating from the data sample what feature vector would correspond to a vowel/A/, that the computation not be distorted by time-warped samples in which the speaker was actually pronouncing a/t/ or a/p/. Of course, this time-alignment of the samples in the data base could be accomplished other ways, including automatic classification of the samples by acoustic segment boundaries according to, e.g., LPC characteristics, but the presently preferred embodiment uses manual classification at this step.

Thus, after this manual classification step, the segment within each sample in the data base which corresponds to the successive acoustic segments which must be included in the reference vector has been established. The average duration of each of the acoustic segments is then computed to establish the number of reference frames needed in each segment. For example, suppose the sibilant /s/ has an average duration of 130 msec. Then, at a reference frame period of 40 msec., three reference frames in the word template will correspond to the sibilant /s/. (The period of the reference frame in the presently preferred embodiment is exactly twice as long as the frame interval in the input speech, for reasons which will be discussed below.)

The next step is to locate, in each of the 100 samples, which portions of the sample shall be included in the computation of the expected characteristics of each of the three reference frames in the word template which are to correspond to the sibilant /s/. That is, in this example the three /s/ reference feature vectors should be computed based on measurements at three points evenly spaced within the duration of the phoneme /s/, for each sample in the data base. Thus, the result of this process is that, for each frame in the word template for which a reference vector must be computed, a unique location within each sample in the data base to correspond to that frame has been identified.

By way of example, FIG. 4 generally illustrates the process by which each reference vector in the word template is computed, based on the corresponding portions of the various samples in the data base. First, a tolerance number called a unanimity factor (nu) is chosen empirically. In the presently preferred embodiment nu is set equal to 0.93, but may be greater or lesser depending on the uniformity of the speakers in the data base, and to some extent on the size of the data base. However, in the presently preferred embodiment, a value greater than 90% is preferably used, and is preferably in the range of 90 to 97%.

The unanimity factor nu tells how many disagreements can be tolerated on any particular bit of the feature vector for corresponding frames before concluding that there is no concurrence of behavior over the population. That is, for example, suppose that nu is chosen equal to 0.93. In this case, if 93 or more of 100 samples in the data base have a value for the first analog parameter (at corresponding frame locations) which is larger than B1L, then the first elements of the reference feature vector and of the mask vector are set equal to 1. If 93 or more of the samples have a value for the first parameter which is below B1L, then the first element of the reference feature vector is 0 and the first element of the mask vector is 1. That is, in this case the population would have agreed that the general behavior is to have the number of zero crossing intervals in the first "bin" less than the threshold B1L. However, if less than 93 samples agree in this respect, then the first element of the mask vector is set equal to zero, and the first element of the reference vector is a "don't care" value and can be 0 or 1.

Thus, this process generates a word template which is a time ordered sequence of vector pairs, namely a feature vector and a mask vector at each reference frame interval.

The basic distance measure, which compares one frame of speech input to some one frame of a word template has been described above. However, the word identification rests not merely on testing the identity of frame to frame, but rather on finding the similarity of a sequence of input frames to the sequence of reference vectors in a word template. In the presently preferred embodiment, a dynamic programming technique is used to find an optimal subsequence match between the sequence of reference vectors in the word template and a subsequence of feature vectors in the speech input. This dynamic programming algorithm permits time-warping by various speakers to be accommodated, but also has other advantages. In particular, the end points can be unconstrained. That is, no separate decision step needs to be made as to which frame in an input sequence of frames the end point of the word template should be identified to. Moreover, a second advantage of this

approach is that the storage requirements are reduced, since the reference frame interval is twice the frame interval imposed on the speech input signal. In general, the unconstrained end-point approach is accomplished by providing a cumulative cost profile, for each point in time, which assumes that the current input frame is the last frame. However, to economize on processor time, the preferred embodiment uses an end-of-word window instead, as will be discussed below.

Thus, the foregoing steps produce a scalar dissimilarity measure D_{Nj} which shows the dissimilarity between an input frame j and a reference frame N . This dissimilarity measure is then transformed, through a dynamic programming procedure, into a minimal subsequence distance measure (scanning error) E_{Nj} , which is preferably defined as follows:

$$E_{Nj} = D_{Nj} + \min \{ E_{N-1, j-1} + K, E_{N-1, j-2} + K/3, E_{N-1, j-4} + K \}$$

The quantity "K" is a constant which is optionally used to impose a warping penalty. That is, the expected ratio of reference frames to sample frames is one reference frame to every two sample frames. However, if this is not in fact the actual spacing, then a penalty amount is added to the minimal subsequence distance for every reference in which the local ratio of input frames to reference frames is different from 2-1. Note that the penalty added where the ratio is locally 3-1 is much smaller than that imposed where the ratio is locally 4-1 or 1-1. Thus, only a modest penalty is added where the input speech is slightly slower than the reference speech rate (down to $1\frac{1}{2}$ times as slow), but a substantially larger penalty is added if the input speech is faster than the reference speech, or is more than $1\frac{1}{2}$ times as slow as the rate affected by the reference speech.

That is, where input frames are matched to reference frames at an average rate which is between 2-1 and 3-1, and where the time distribution of the input frame is the same as that of the reference frame, then the particular mappings of reference frame onto input frame within the optimal subsequence will vary between every other input frame and every third input frame, and the total speed-mismatch penalty will be a linear function of the speech rate mismatch. However, where the warping of the input sample is sufficiently nonlinear that, within the optimal subsequence, some adjacent pairs of the reference template sequence match either adjacent input frames or to input frames which are separated by three other input frames, an additional penalty will be added to the smooth penalty for linear warping. This additional penalty may be referred to as a nonlinear warping penalty, although it should be noted that nonlinear warping is penalized only if it causes some local portion of the reference-to-input mapping to be denser than 1-2 or sparser than 1-3. Thus, this warping penalty incorporates speech-rate information into the recognition process, but does not require large additional amounts of computation time.

The warping penalty is optional, and is not strictly necessary for practicing the present invention. That is, the iterative statement of the dynamic programming measure can be restated as

$$E_{Nj} = D_{Nj} + \min_{k \in [1,4]} \{E_{N-1,j-k}\}$$

The presently preferred embodiment does not use warping penalties to minimize the computational load.

Alternatively, a larger than 2-to-1 warping factor can be permitted, or a sparser ratio of reference templates to input frames could be used, as desired. The warping penalties can accordingly be widely varied.

The foregoing dynamic programming procedure can provide a cumulative fit measure for each word in the vocabulary, at each input frame interval. In this case, the recognizer is capable of operating in a connected-speech recognition mode rather than an isolated-speech recognition mode.

However, this imposes a heavy additional processing load and is not the preferred embodiment of the invention.

That is, the processing load required to find a cumulative optimal subsequence match at each input frame interval is too much for the economical implementations at which the present invention is especially directed. To reduce this processing load, words are preferably recognized only at word ending points identified by an end-of-word detector. The operation of the end-of-word detector will now be described with reference to FIG. 3. The end-of-word operation as depicted in FIG. 3 provides the integer feature vector 32 for each frame of input speech as an input to a plural frame buffer memory 34, which may include storage for 20 frames of speech data for example.

In the presently preferred embodiment, the zero crossings are not only sorted into bins, but a count is kept of the total number of zero crossings. For example, this can be done by adding together the counts in the various bins of the integer feature vector, depending on the bin threshold values. Alternatively, this can be done by simply keeping a direct running count of the number of zero crossings, and holding this as a directly computed parameter during each frame of input speech. A further alternative is simply to count the number of high-frequency zero crossings for each frame, and sum those across frames as at 36.

The key test which is implemented in the end-of-word decision of this aspect of the present invention is to ascertain whether the number of zero crossings exceeds a given threshold number as at 38 during a reasonably long period of time (e.g. 300 milliseconds). If the threshold number is exceeded as at 40, this large number of zero crossings indicates that no low-frequency energy, and therefore presumably no speech, is present during this 300 millisecond window. It should be noted that this is somewhat sensitive to the bias level used in the Schmitt trigger (or other center-clipping mechanism). That is, if the bias level in the Schmitt trigger is set too high, then noise at the end of a word, in a quiet environment, will not be able to produce the high number of zero crossings required for the detection of end of word. Correspondingly, if the bias level is too low, a long unvoiced consonant (such as the s at the end of a word such as "guess") may generate enough high-frequency zero crossings to trigger the end of word detector erroneously.

Thus, the end-of-word detector selectively indicates that an end-of-word has occurred. If so, then word recognition is performed on the assumption that the

word will have ended during a second window period, which is not necessarily the same window over which the end-of-word operates. That is, in the presently preferred embodiment, an end-of-word is detected when 300 milliseconds have occurred without input speech energy, and the first 200 milliseconds of the 300 millisecond end-of-word detection window are then searched for a hypothetical word ending point. However, this second window, during which an end-of-word is looked for, can be the same as or different from the end-of-word detection window, and can be varied within very broad parameters. In the example shown in FIG. 3 the end-of-word detection lies within the first 13 frames of speech data included in the 20-frame buffer memory 34. The essential trade-off here is that, if the recognition window is made smaller, the processor load is reduced but the frequency of non-recognition errors is likely to be increased.

The invention as presently practiced is embodied in a VAX 11/780, with analog input and output connections (i.e., microphone, preamplifier, analog-to-digital converter, digital-to-analog converter, audio amplifier and loudspeaker), and is implemented in the Fortran code in the attached appendix which is hereby incorporated by reference. However, as discussed above, the present invention can be implemented in a cheap microcomputer system, and the contemplated best modes of the invention in the future are expected to be microprocessor or microcomputer embodiments.

In particular, an embodiment of the present invention in an 8-bit microprocessor system is believed to be straight-forward. No expensive data converter chip or means for energy measurement is required. The only analog stages needed, in the preferred embodiment are the low-pass filter, differentiator, and Schmitt trigger. If the present invention is embodied in a 16-bit system, the additional processing power and word length will mean simply that a slightly larger vocabulary can be accommodated, and will also make development of the vocabulary templates slightly easier.

As will be obvious to those skilled in the art, the present invention provides numerous broad points of novelty over the prior art of speech recognition. Therefore, the scope of the present invention can be embodied in numerous modifications and variations and is not limited as specified in the accompanying claims.

What is claimed is:

1. A method for recognizing speech independent of the speaker thereof, said method comprising:

receiving an analog input speech signal;
conditioning said analog speech signal to produce a sequence of rectangular waveforms of polarity signs alternating between plus and minus polarities as a digital waveform signal;

counting the number of polarity transitions in the digital waveform signal to obtain a zero-crossing count for each frame of the digital waveform signal;

measuring the time duration intervals between zero-crossings of the digital waveform signal;

providing a sequence of binary feature vectors based upon the measurements of the time duration intervals between zero-crossings of the digital waveform signal and corresponding to respective frames of the digital waveform signal;

providing a vocabulary consisting of a relatively small number of words, wherein each of the words

included in the vocabulary is represented by a plurality of binary reference vectors which have been organized in sequences with each of said binary reference vector sequences corresponding to a word acoustically distinct from the other words included in the vocabulary;

comparing each of said binary feature vectors with each of said plurality of binary reference vectors; determining a distance measure with respect to each of said binary reference vectors for each successive binary feature vector in said sequence of binary feature vectors in response to the comparison therebetween; and

recognizing words in accordance with the distance measures between each of said binary reference vector sequences and successively received binary feature vectors corresponding to respective frames of the digital waveform signal.

2. A method for recognizing speech as set forth in claim 1, wherein the provision of said sequence of binary feature vectors is accomplished by sorting the zero-crossing time duration interval measurements received during respective frames of the digital waveform signal into corresponding ones of a plurality of bins respectively representative of different time duration intervals between zero-crossings;

counting the number of zero-crossing time duration intervals for each of the plurality of bins;

comparing the counts of respective bins to upper and lower reference thresholds corresponding to the respective bins; and

providing said sequence of binary feature vectors in response to the comparison between the counts of the respective bins and the upper and lower thresholds corresponding thereto.

3. A method for recognizing speech as set forth in claim 1, further including

establishing the identity of an end of word prior to the recognition of a word as a precondition thereto, the establishing of said end of word identification including:

monitoring the zero-crossing count for the digital waveform signal, and

declaring an end of word condition whenever the average frequency of said zero-crossings exceeds an end point target zero-crossing frequency for a time duration longer than a predetermined reference time duration.

4. The method of claim 1, wherein said distance measure-determining step comprises a Hamming distance measurement.

5. The method of claim 3, wherein said distance measure-determining step comprises a Hamming distance measurement.

6. The method of claim 1, wherein said recognizing step comprises a dynamic programming step to achieve an optimal subsequence match between one of said sequences of said binary reference vectors and spaced successive ones of said binary feature vectors.

7. The method of claim 3, wherein said recognizing step comprises a dynamic programming step to achieve an optimal subsequence match between one of said sequences of said binary reference vectors and spaced successive ones of said binary feature vectors.

8. The method of claim 4, wherein said recognizing step comprises a dynamic programming step to achieve an optimal subsequence match between one of said

sequences of said binary reference vectors and spaced successive ones of said binary feature vectors.

9. The method of claim 5, wherein said recognizing step comprises a dynamic programming step to achieve an optimal subsequence match between one of said sequences of said binary reference vectors and spaced successive ones of said binary feature vectors.

10. The method of claim 1, wherein said conditioning step includes center clipping said analog input speech signal.

11. The method of claim 3, wherein said conditioning step includes center clipping said analog input speech signal.

12. The method of claim 10, wherein said center clipping step is performed by a Schmitt trigger.

13. The method of claim 11, wherein said center clipping step is performed by a Schmitt trigger.

14. The method of claim 1, wherein said conditioning step includes the performance of an operation corresponding to differentiation of said analog input speech signal.

15. The method of claim 3, wherein said conditioning step includes the performance of an operation corresponding to differentiation of said analog input speech signal.

16. A word recognition system for identifying a spoken word independent of the speaker thereof, wherein the spoken word is represented by an analog speech signal, said word recognition system comprising:

signal conditioning means for receiving an analog input speech signal and producing a digital waveform signal as a sequence of rectangular waveforms of polarity signs alternating between plus and minus polarities, said signal conditioning means including a zero-crossing detector for counting the number of polarity transitions in the digital waveform signal to obtain a zero-crossing count for each frame of the digital waveform signal;

memory means storing a plurality of binary reference templates of digital speech data respectively representative of individual words and comprising the vocabulary of the word recognition system, the vocabulary consisting of a relatively small number of words with each of the words included in the vocabulary being represented by a binary reference template defined by a predetermined plurality of binary reference vectors arranged in a predetermined sequence and comprising an acoustic description of an individual word in a time-ordered sequence, each of said binary reference templates corresponding to a word acoustically distinct from the other words included in the vocabulary;

means operably coupled to said signal conditioning means for extracting binary feature vectors from said digital waveform signal based upon the time duration intervals between zero-crossings of the digital waveform signal;

means operably associated with said binary feature vector extracting means for comparing each binary feature vector of said digital waveform signal with the corresponding binary reference vectors of each of said binary reference templates to provide a distance measure with respect to each of the binary feature vectors and the predetermined binary reference vector sequences defining acoustic descriptions of the respective words included in the vocabulary of the word recognition system; and

17

word recognizing means for determining which one of the plurality of the binary reference templates is the closest match to said digital waveform signal representing said analog input speech signal based upon the distance measures of each of said binary reference vector sequences and successively received binary feature vectors corresponding to respective frames of the digital waveform signal.

17. A word recognition system as set forth in claim 16, further including dynamic programming means operably connected to the output of said comparing means for receiving the distance measures between each of said binary reference vector sequences and successively received binary feature vectors to provide an optimal subsequence match therebetween.

18. A word recognition system as set forth in claim 16, further including word-end detector means operably

18

interposed between said zero-crossing detector of said signal conditioning means and said binary feature vector extracting means for monitoring the zero-crossing count for the digital waveform signal and producing a signal output declaring an end of word condition whenever the average frequency of said zero-crossings exceeds an end point target zero-crossing frequency for a time duration longer than a predetermined reference time duration; and

said word recognizing means including decision logic means having inputs for receiving the distance measures of each of said binary reference vector sequences and successively received binary feature vectors and the output from said word-end detector means as a precondition to providing a word recognition output.

* * * * *

20

25

30

35

40

45

50

55

60

65

United States Patent [19][11] **Patent Number:** **5,475,792****Stanford et al.**[45] **Date of Patent:** **Dec. 12, 1995****[54] TELEPHONY CHANNEL SIMULATOR FOR
SPEECH RECOGNITION APPLICATION****[75] Inventors:** Vince M. Stanford, Gaithersburg;
Norman F. Brickman, Potomac, both
of Md.**[73] Assignee:** International Business Machines
Corporation, Armonk, N.Y.

4,856,066	8/1989	Lemelson	381/36
4,897,878	1/1990	Boll et al.	381/43
4,905,286	2/1990	Sedgwick et al.	381/43
4,922,538	5/1990	Tchorzewski	381/42
4,933,973	6/1990	Porter	381/43
4,956,871	9/1990	Swaminathan	381/31
5,027,406	6/1991	Roberts et al.	381/43
5,036,538	7/1991	Oken et al.	381/43
5,068,899	11/1991	Ellis et al.	381/31
5,105,463	4/1992	Veldhuis et al.	381/30

[21] Appl. No.: 201,157**[22] Filed:** Feb. 24, 1994**FOREIGN PATENT DOCUMENTS**

215573 8/1985 European Pat. Off. .

Related U.S. Application Data**[63]** Continuation of Ser. No. 948,031, Sep. 21, 1992, abandoned.**[51] Int. Cl.⁶** G01L 9/00**[52] U.S. Cl.** 395/2.42; 395/2.63; 395/2.65;
395/2.36**[58] Field of Search** 381/29-45; 395/2.1,
395/2.2-2.25, 2.35-2.37, 2.4-2.65; 375/122,
25, 27**[56] References Cited****U.S. PATENT DOCUMENTS**

4,069,393	1/1978	Martin et al.	179/1 SD
4,087,630	5/1978	Browning et al.	179/1 SD
4,144,582	3/1979	Hyatt	364/900
4,461,024	7/1984	Rengger et al.	381/46
4,481,593	11/1984	Bahler	381/43
4,489,435	12/1984	Moshier	381/43
4,718,088	1/1988	Baker et al.	381/43
4,720,802	1/1988	Damoulakis et al.	381/43
4,783,803	11/1988	Baker et al.	381/42
4,805,218	2/1989	Bamberg et al.	381/43
4,829,572	5/1989	Kong	381/41

OTHER PUBLICATIONS

Takebayashi et al., "Telephone Speech Recognition Using A Hybrid Method", ICPR (International Conference of Pattern Recognition), Dec. 1989, pp. 1232-1235.

IEEE Article by K. F. Lee & H. W. Hon, "Large Vocabulary Speaker Independent Continuous Speech Recognition Using HMM," 1988, pp. 123-126, (CH2561-9/88/00000-0123).

Primary Examiner—Allen R. MacDonald*Assistant Examiner*—Michelle Doerrler*Attorney, Agent, or Firm*—John E. Hoel; Martin J. McKinley**[57] ABSTRACT**

A telephony channel simulation process is disclosed for training a speech recognizer to respond to speech obtained from telephone systems. An input speech data set is provided to a speech recognition training processor, whose bandwidth is higher than a telephone bandwidth. The process performs a series of alterations to the input speech data set to obtain a modified speech data set. The modified speech data set enables the speech recognition processor to perform speech recognition on voice signals from a telephone system.

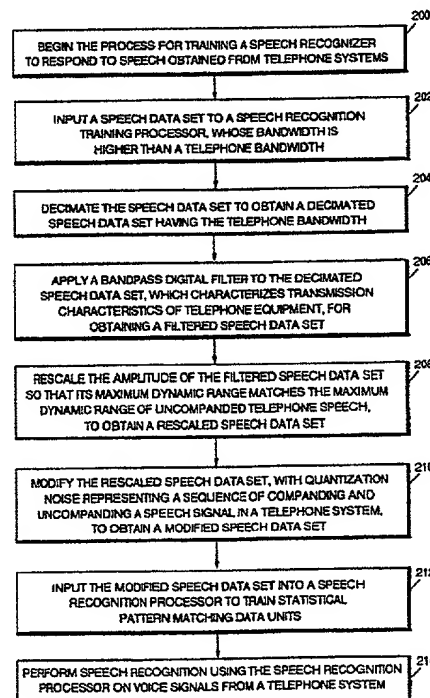
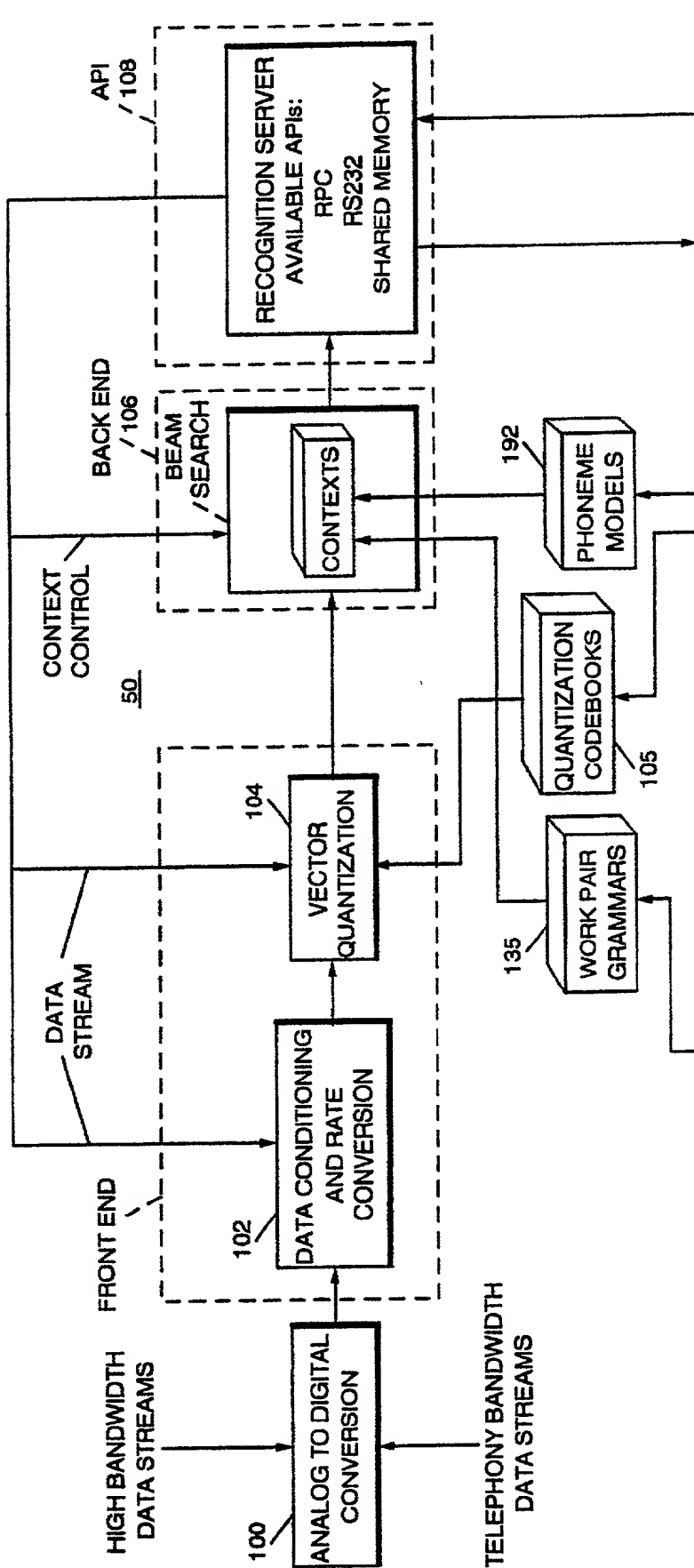
7 Claims, 7 Drawing Sheets

FIG. 1A



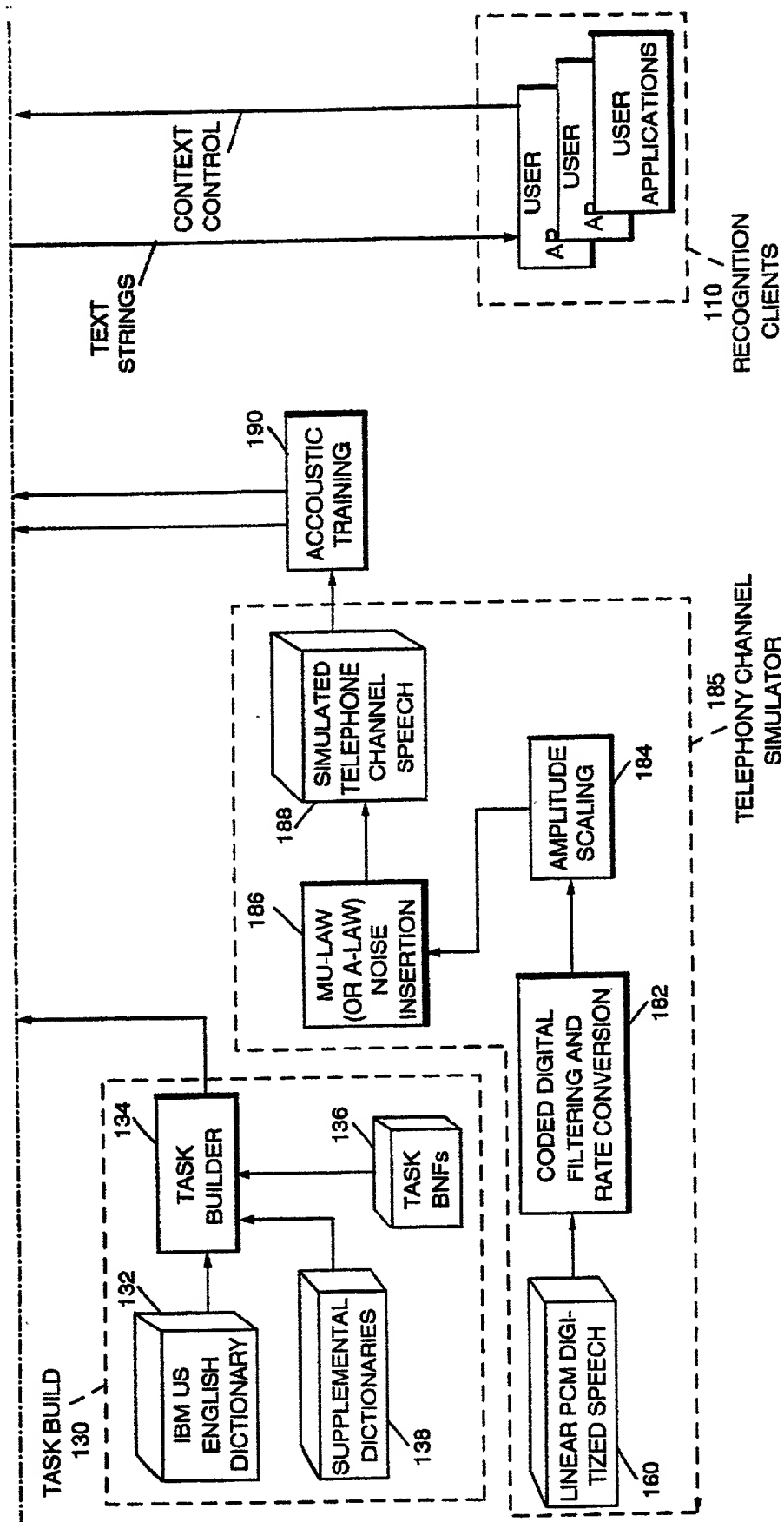


FIG. 1B

FIG. 2

READ THE NUMBER OF FILTER WEIGHTS
FROM THE FILTER SIZE FILE, AND THE
WEIGHTS FROM THE ASSOCIATED
WEIGHT FILE.

$N = 295$

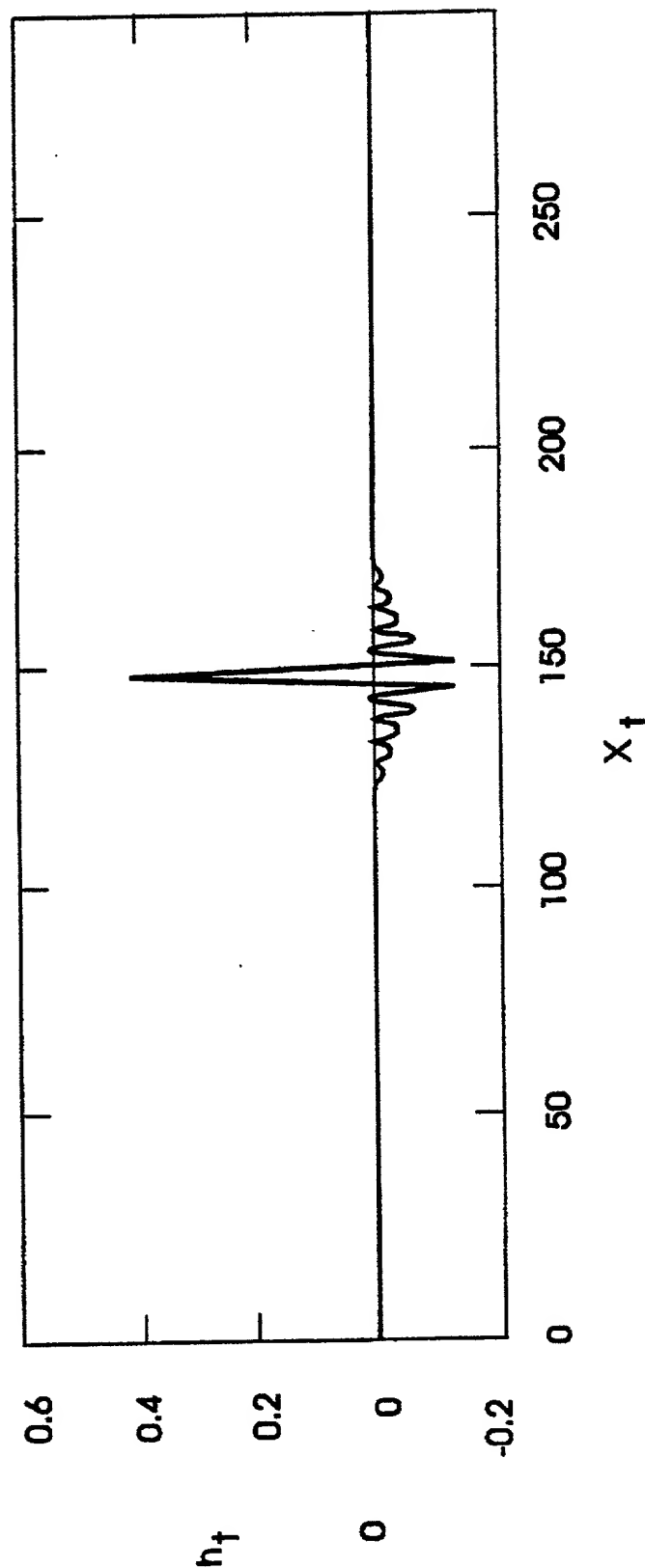
$N = \text{READ}(\text{FILTERSIZEFILE})$

$t = 0 \dots N-1$

$x_t = t$

$h_t = \text{READ}(\text{FILTERWEIGHTFILE})$

READ IMPULSE RESPONSE



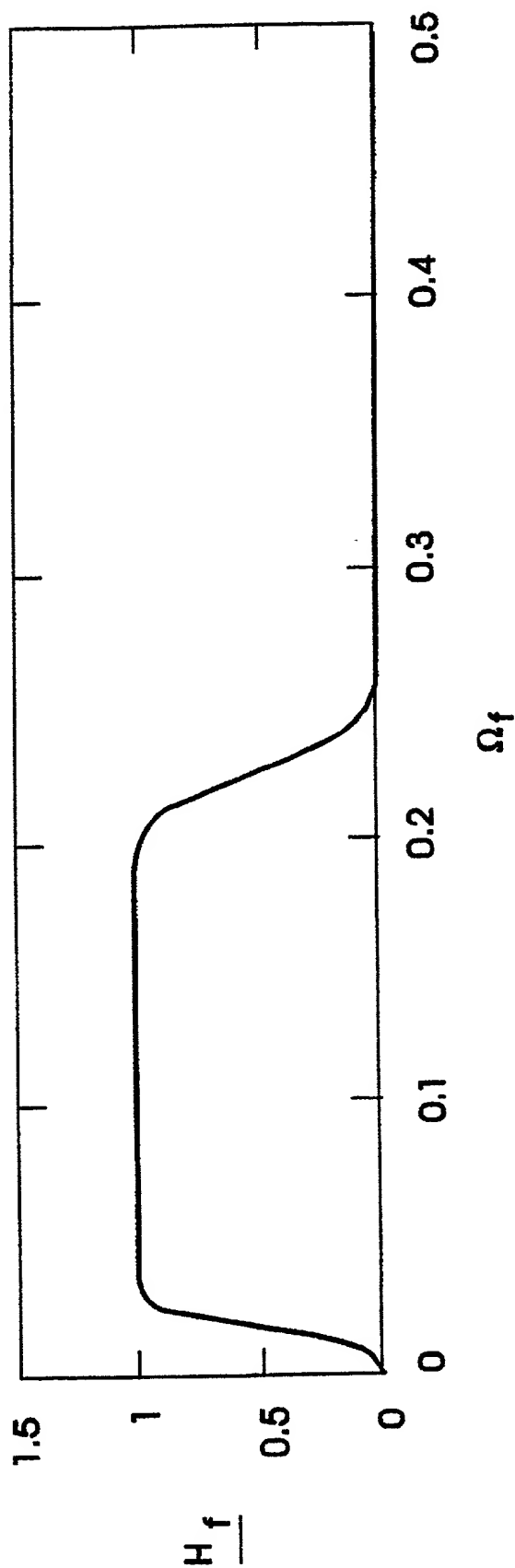
TELEPHONIC CODEC FILTER IMPULSE RESPONSE

FIG. 3

$F = 256$
 $f = 0.1F$
 $\Omega_f = \frac{f}{2F}$

$$H_f = \left| \sum_{h_f}^{-1} 2 \times 1 \frac{f}{2F} \right|$$

FILTER MAGNITUDE RESPONSE FUNCTION



MAGNITUDE RESPONSE VERSUS NORMALIZED RADIAN FREQUENCY

FIG. 4

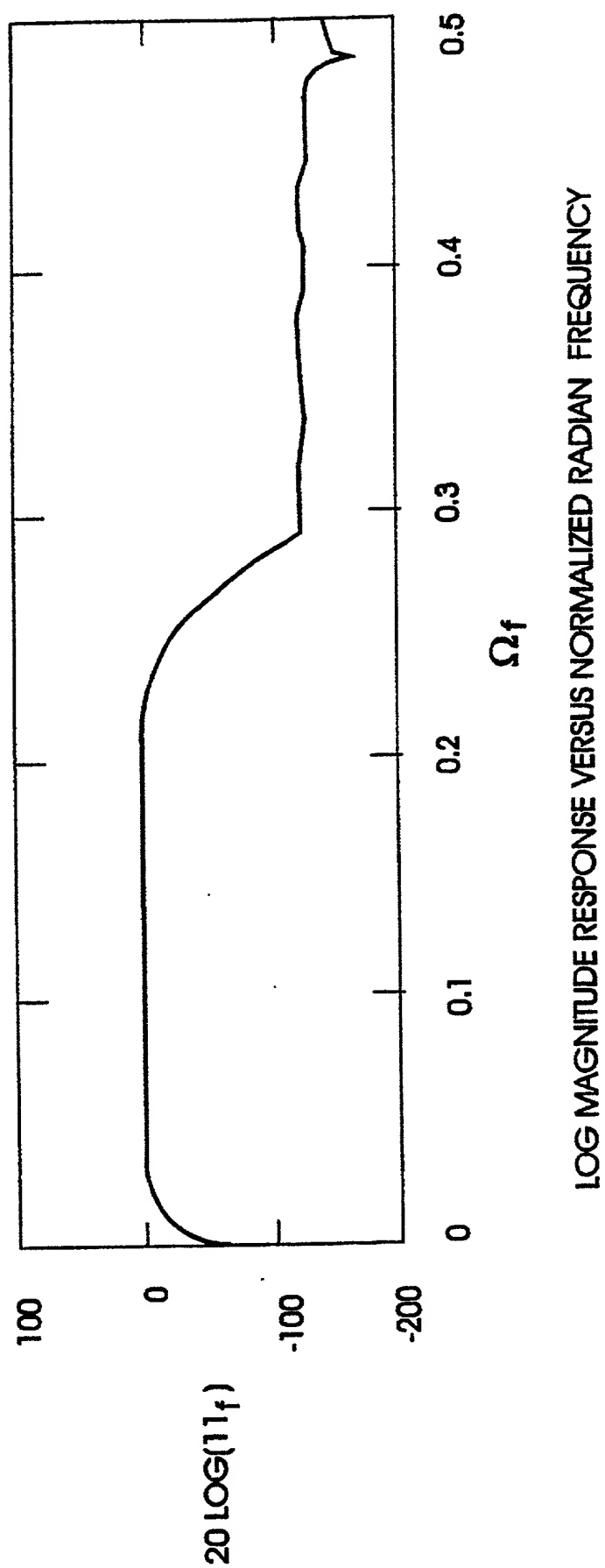


FIG. 5

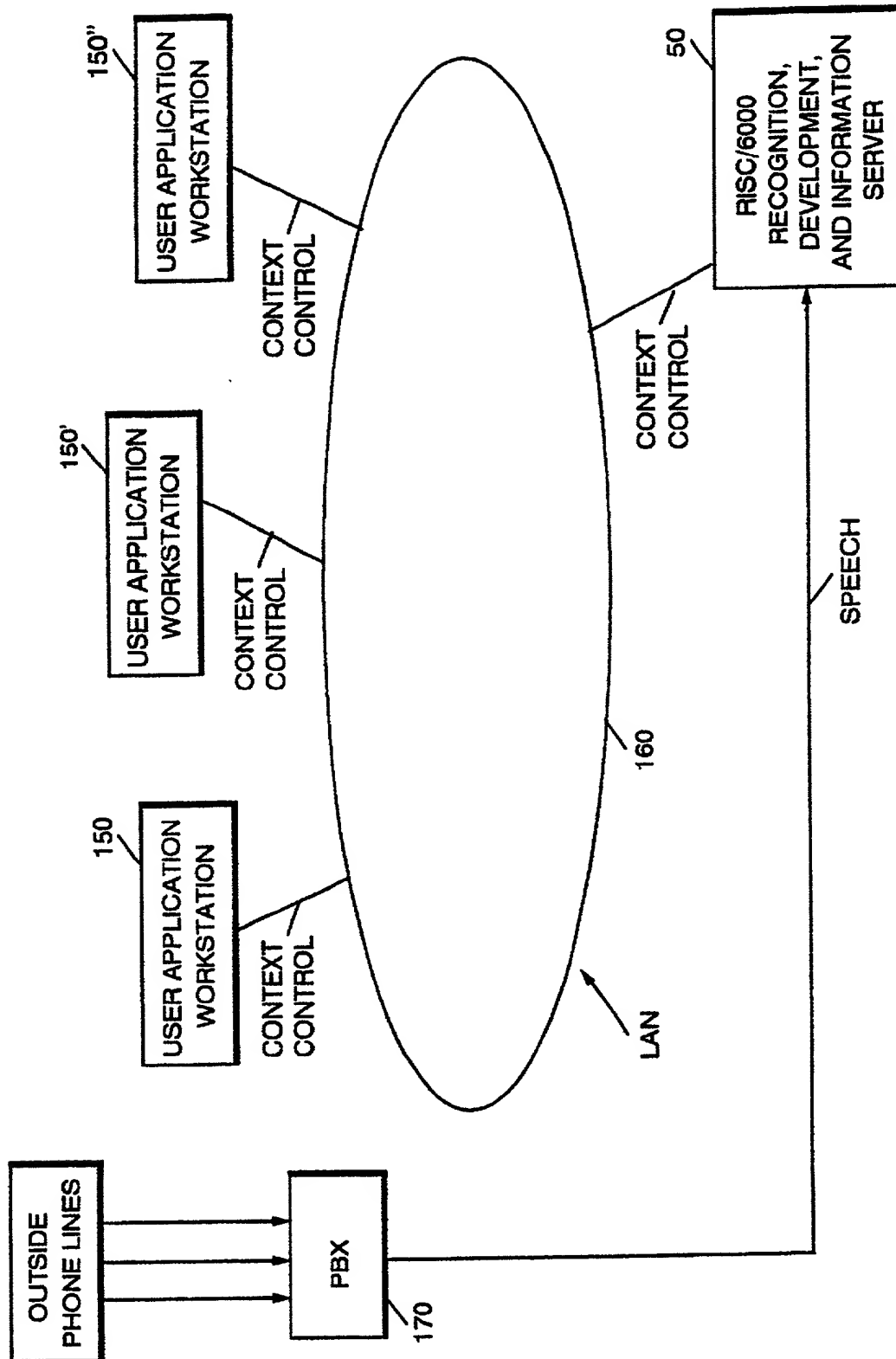
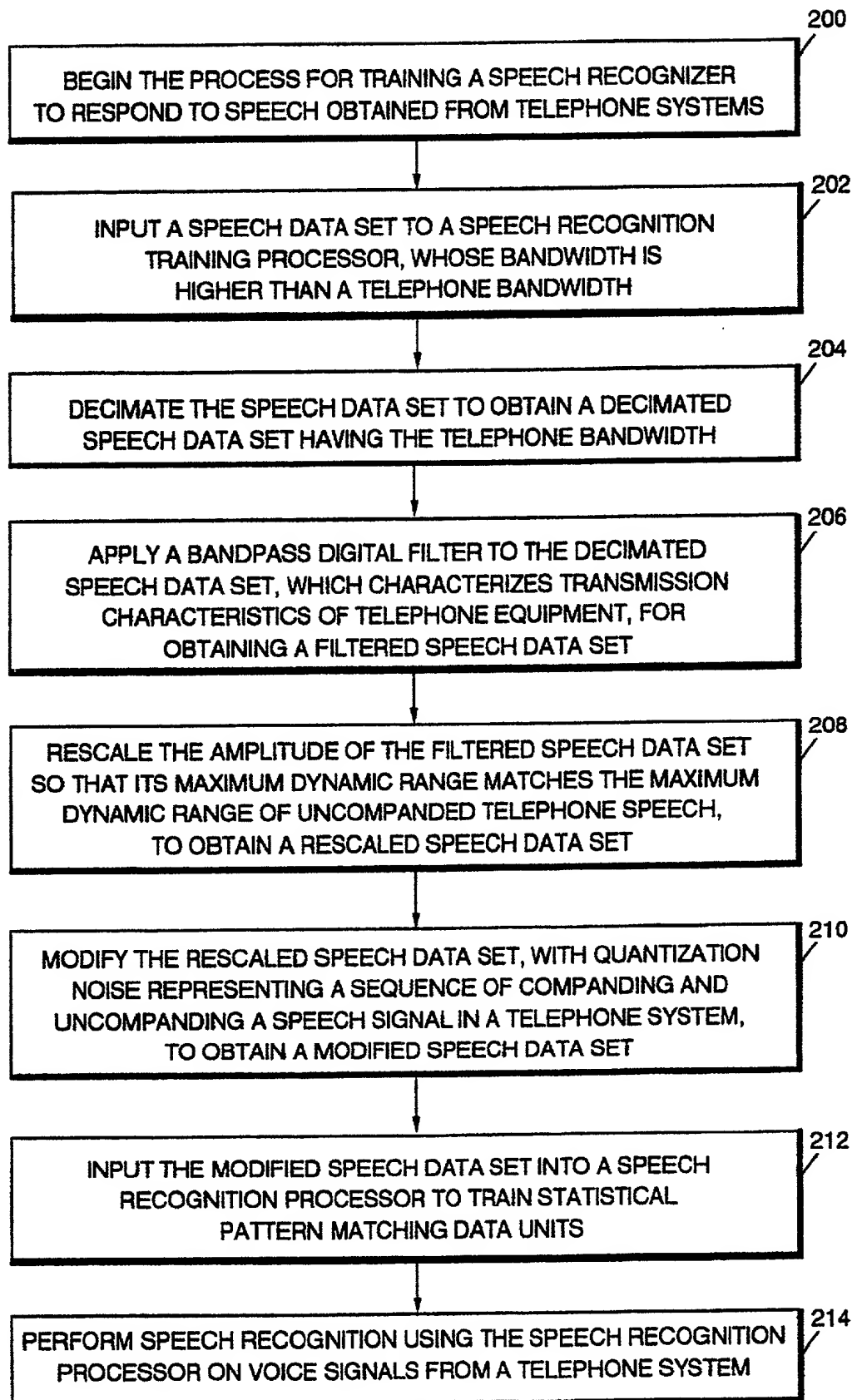


FIG. 6



TELEPHONY CHANNEL SIMULATOR FOR SPEECH RECOGNITION APPLICATION

This is a continuation of prior application Ser. No. 07/948,031, filed Sep. 21, 1992, now abandoned.

BACKGROUND OF THE INVENTION

This invention relates adapting a speech recognition system to be capable of operating over telephonic public switched networks.

Speech recognition systems are well known to the art. Examples include the IBM Tangora ("A Maximum Likelihood Approach to Continuous Speech Recognition;" L. R. Bahl, F. Jelinek, R. Mercer; Readings in Speech Recognition; Ed.: A. Waibel, K. Lee; Morgan Kaufmann, 1990; pp. 308-319.) and Dragon Systems Dragon 30k dictation systems. Typically, they are single user, and speaker-dependent. This requires each speaker to train the speech recognizer with his or her voice patterns, during a process called "enrollment". The systems then maintain a profile for each speaker, who must identify themselves to the system in future recognition sessions. Typically speakers enroll via a local microphone in a low noise environment, speaking to the single machine on which the recognizer is resident. During the course of enrollment, the speaker will be required to read a lengthy set of transcripts, so that the system can adjust itself to the peculiarities of each particular speaker.

Discrete dictation systems, such as the two mentioned above, require speakers to form each word in a halting and unnatural manner, pausing, between, each, word. This allows the speech recognizer to identify the voice pattern associated each individual word by using preceding, and following, silences to bound the words. The speech recognizer will typically have a single application for which it is trained, operating on the single machine, such as Office Correspondence in the case of the IBM Tangora System.

Multi-user environments with speaker dependent speech recognizers require each speaker to undertake tedious training of the recognizer for it to understand his or her voice patterns. While it has been suggested that the templates which store the voice patterns may be located in a common database wherein the system knows which template to use for a speech recognition by the speaker telephone extension, each speaker must none-the-less train the system before use. A user new to the system calling from an outside telephone line will find this procedure to be unacceptable. Also, the successful telephonic speech recognizer will be capable of rapid context switches to allow speech related to various subject areas to be accurately recognized. For example, a system trained for general Office Correspondence will perform poorly when presented with strings of digits.

The Sphinx system, first described in the Ph.D Dissertation of Kai-Fu Lee ("Large Vocabulary Speaker and Dependent Continuous Speech Recognition: The Sphinx System;" Kai-Fu Lee; Carnegie Mellon University, Department of Electrical and Computer Engineering; April 1988; CMU-CS-88-148), represented a major advance over previous speaker, dependent recognition systems in that it was both speaker independent, and capable of recognizing words from a continuous stream of conversational speech. This system required no individualized speaker enrollment prior to effective use. Some speaker dependent systems require speakers to be reenrolled every four to six weeks, and require users to carry a personalized plug-in cartridge to be understood by the system. Also with continuous speech

recognition, no pauses between words are required, thus the Sphinx system represents a much more user friendly approach to the casual user of a speech recognition system. This will be an essential feature of telephonic speech recognition systems, since the users will have no training in how to adjust their speech for the benefit of the recognizer.

A speech recognition system must also offer real time operation with a given modest vocabulary. However, the Sphinx System still had some of the disadvantages of the prior speaker dependent recognizers in that it was programmed to operate on a single machine in a low noise environment using a microphone and a relatively constrained vocabulary. It was not designed for multi-user support, at least with respect to the different locations, and multiple vocabularies for recognition.

This invention overcomes many of the disadvantages of the prior art.

OBJECTS OF THE INVENTION

It is therefore an object of the present invention to provide a continuous speech speaker independent speech recognizer suitable for use with telephony equipment with input from speakers, both local and long distance.

It is another object of the invention to train the system from a vocabulary gathered in low noise conditions to recognize speech patterns in a high noise e.g., telephone environment.

It is another object of the invention to enable a plurality of voice applications to be recognized by a speech recognizer concurrently in a computer network or telephonically.

SUMMARY OF THE INVENTION

These and other objects are accomplished by speech recognition systems, architected on a client/server basis on a local area or wide area network. The speech recognition system is divided into a number of modules including a front end which converts the analog or digital speech data into a set of Cepstrum coefficients and vector quantization values which represent the speech. A back end uses the vector quantization values and recognizes the words according to phoneme models and word pair grammars as well as the context in which the speech made. By dividing the vocabulary into a series of contexts, situations in which certain words are anticipated by the system, a much larger vocabulary can be accommodated with minimum memory. As the user progresses through the speech recognition task, contexts are rapidly switched from a common database (see the Brickman, et al. Patent Application cited herein). The system also includes an interface between a plurality of user applications also in the computer network.

The system includes training modules, training and task build modules to train the system and to build the word pair grammars for the context respectively.

The invention includes a telephony channel simulation process for training a speech recognizer to respond to speech obtained from telephone systems. The method begins by inputting a data set to a speech recognition training processor, whose bandwidth is higher than a telephone bandwidth. Then, the speech data set is decimated to obtain a decimated speech data set having the telephone bandwidth. Then, a bandpass digital filter is applied to the decimated speech data set, which characterizes transmission characteristics of telephone equipment. This is done to obtain a filtered speech data set. Then, the amplitude of the filtered speech data set

is rescaled, so that its maximum dynamic range matches the maximum range of unexpanded telephone speech. This is done to obtain a rescaled speech data set. Then, the rescaled speech data set is modified with quantization noise representing a sequence of companding and unexpanding a speech signal in a telephone system. This is done to obtain a modified speech data set. Then, the modified speech data set is input to a speech recognition processor, to train statistical pattern matching data units. The method results in the speech recognition processor being able to perform speech recognition on voice signals from a telephone system.

DESCRIPTION OF THE FIGURES

These and other objects, features and advantages will be more fully appreciated with reference to the accompanied Figures.

FIG. 1 illustrates the logical architecture of a continuous speech recognition system, which includes the telephony channel simulator invention.

FIG. 2 is a graph which characterizes the telephonic codec filter impulse response.

FIG. 3 is a graph illustrating the magnitude response verses normalized radian frequency.

FIG. 4 is a graph which illustrates the log magnitude response verses normalized radian frequency.

FIG. 5 is a block diagram of a network for a recognition server operating in a telephony customer service call center.

FIG. 6 is a flow diagram of a sequence of operational steps for a process for training a speech recognizer to respond to speech obtained from telephone systems.

DETAILED DESCRIPTION OF THE INVENTION

The bandwidth reductions and noise introduced by telephone lines reduce the accuracy of all speech recognition systems. This effect increases with the size of the vocabulary that must be recognized at each moment in time. The use of rapidly switchable speech recognition contexts useful to this invention, so that individual contexts can be limited in size. Context switching is described in the copending U.S. patent application Ser. No. 07/947,634, filed Sep. 21, 1992, by N. F. Brickman, et al., entitled "Instantaneous Context Switching For Speech Recognition Systems" assigned to the IBM Corporation and incorporated herein by reference.

FIG. 1 illustrates the logical architecture of the IBM Continuous Speech Recognition System (ICSRS) independent of hardware configurations. At a broad level, ICSRS consists of components addressing the following areas:

Data Acquisition—Data are converted in block 100 from analog to digital form, or potentially demultiplexed from other channels in the case of telephonic data.

Data Compression—The ICSRS Front End blocks 102 and 104, conditions, resamples, and compresses speech data streams to 300 bytes per second during the vector quantization step.

Speech Recognition—The Back End 106 performs the actual speech recognition by pattern matching phoneme models 192 using a grammar-guided beam search algorithm. The phoneme models 192 and word pair grammars 135 together constitute the recognition contexts. Single or multiple instances of Back-End recognizers can be deployed either remotely or locally to the Front-End instances which acquire and compress the speech data.

Task Building—The task building component 130 allows the construction of recognition contexts off-line, compiles the word pair grammars for use at run time, and binds appropriate phoneme models to the task (context).

Application Program Interface—The API 108 offers RPC based recognition services which allow data stream control, context loading, and activation.

Telephone Channel Simulator—the simulator 185 connects high bandwidth, high resolution speech data sets into phoneme models 192 and telephone speech, having the reduced sampling rate, compressed bandwidth and compressed dynamic range of telephone speech.

During speech recognition, either a high bandwidth voice data stream from a local microphone or a low bandwidth voice data stream, such as would be associated with telephony, is received by the Analog to Digital Conversion block 100. The Analog to Digital Conversion 100 can be performed by a hardware card such as the IBM M-Audio Capture and Playback Card (M-ACPA) card in the voice workstation. It has a digital signal processor which processes either the high bandwidth or telephony bandwidth signals and converts them to a series of digitally sampled data points. This conversion could also be performed by a digital PBX, and the telephony data streams provided in 8 KHz, 8-bit mu-law/a-law compressed format.

For purposes of the present invention, high bandwidth is defined as being a sampling rate of 16 kilohertz or above. Low bandwidth is defined as 8 kilohertz or below which is what the general telephone system in the United States uses for digital voice. The A/D conversion block 100 is optional as in a telephone system the digital information could come in from a private phone exchange (PBX).

The first major block in the "front end" for speech recognition is the Data Conditioning and Rate Conversion (DCRC) block 102. The digitalized input from the A/D conversion 100 is at 44 or 8 kilohertz. A resampling technique referenced to herein as decimation, is used as provided by the public literature in the IEEE (A General Program to Perform Sampling Rate Conversion of Data by Rational Ratios," from "Programs for Digital Signal Processing," Ed.: Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Processing Society; IEEE Press, 1979; Section 8.2, pp. 8.2-1 to 8.2-7 by R. E. Crochiere). The DCRC 102 samples and uses anti-aliasing filters on the digitized signal to create either a 16 kilohertz or 8 kilohertz data stream, for subsequent use. Both the DCRC and Vector Quantization processes are described in greater detail below.

After data conditioning and rate conversion in speech recognition, the voice data is passed to the Vector Quantization block 104. In Vector Quantization, the digital data stream is segmented into Frames of one-fiftieth of a second duration, resulting in 320, 220, and 160 samples each at 16 KHz, 11 KHz, and 8 KHz sampling rates respectively. In one preferred embodiment, there are a hundred frames per second computed from any bandwidth speech signal, they are then over-lapped by fifty-percent, and have a Hamming window applied. The Hamming window is well defined in the public literature ("Theory and Application of Digital Signal Processing," L. R. Rabiner, B. Gold; Prentice Hall, 1975, pp. 91).

After the voice data stream is broken into frames, the vector quantization step extracts features from each frame. In the extraction portion of the Vector quantization step, a series of parameters called the LPC Cepstrum Coefficients are calculated. The Cepstrum Coefficients extract and sum-

marize, some of the important characteristics of the speech for pattern recognition. In each frame of data, a fiftieth of a second of speech is encapsulated. One would expect to have fifty frames per second, however, there is fifty-percent overlap so a hundred frames per second are generated. To calculate the Cepstrum Coefficients, first a Hamming window, which is a cosine bell, is applied to the voice data. A Hamming window tapers the edges each frame of voice data to make the data extracted behave more like they would in an infinite duration continuous Fourier Transform.

The Hamming windowed frames are pre-filtered using a filter whose z-transform is $1.0-0.97 \cdot z^{-1}$, ("Large Vocabulary Speaker and Dependent Continuous Speech Recognition: The Sphinx System," Fai-Fu Lee; Carnegie Mellon University, Department of Electrical and computer engineering; April 1988; CMU-CU-88-148) page 49 in order to flatten the speech spectrum. Then 14 auto-correlation coefficients are computed. The auto-correlation coefficients are used to compute the Cepstrum coefficients in a manner well known in the public literature, described in ("Digital Processing of Speech signals," Prentice Hall Signal Processing Series; 1978, pp. 401-402, 411-413). Thirteen Cepstral coefficients are derived from the 14 auto-correlation coefficients. Other numbers of auto-correlation coefficients and dimensions of numbers of Cepstrum coefficients are possible. The statistical properties of these coefficients are used to guide the final vector quantization step.

Vector quantization is also used in the training process 190. The adjustment of the training data described below are crucial in enabling the base Sphinx recognition engine to operate over telephony equipment, and hence to the invention described herein. In the training process 190, a number of sentences are taken, currently between ten to fifteen thousand, and segmented into frames, from which auto-correlation and Cepstrum coefficients are calculated. A clustering procedure is applied to segregate the Cepstrum frame features into two hundred and fifty six classes using a k-means type clustering procedure, described in ("An Algorithm for Vector Quantizer Design," Y. Linde, A. Buzo, R. Gray, IEEE Transactions on Communications, Vol. Com-28, No. 1, January 1980). The centers of these Cepstrum clusters, and their class labels, taken together, are hereafter referred to as "code books". The quantization code books 105 stores the code book for telephone speech, generated by acoustic training functions 190. It will store a second code book for high bandwidth speech.

For the final step of vector quantization, block 104 refers to a code book in the quantization code books 105, FIG. 1, derived in the training procedure, just described, to determine which cluster center is closest to the frame Cepstral coefficients. The current frame is then assigned to the class represented by that code book value. Since there are 256 classes, the VQ value is represented by one byte. There are two other one-byte VQ values derived, from the differential Cepstrum, and the power in the frame. There are three one-byte VQ values derived one hundred times per second, resulting in a compression of the speech data stream to 2,400 bits per second.

Part of the telephony invention herein described is that a completely different code book, which characterizes the speech for the recognizer, must be derived for the telephony data and stored in the quantization code books 105 of FIG. 1. Another part of the invention is that a corresponding phoneme model must be derived for the telephony data and stored in phoneme models 192. A telephone appreciably changes the speech signal, because of sampling rate reductions, bandwidth compression, and dynamic range compression.

However, rather than using voice samples gathered over the telephone, which involves a significant work effort, high bandwidth samples can be processed to simulate the telephone channel characteristics. This allows using the large, readily available speech data files used in the initial training of the Sphinx System, to enable telephonic speech recognition. The telephone channel simulator is the invention described here.

The telephone channel simulation is accomplished in a three phased process as follows:

1.) Conversion to Telephone Bandwidth

High bandwidth, high resolution speech data set, as provided by references, ("Speech Corpora Produced on CD-ROM Media by The National Institute of Standards and Technology (NIST)," April 1991; "DARPA Resource Management Continuous Speech Database (RMI) Speaker Dependent Training Data," September 1989 NIST Speech Discs 2-1.1, 2-2.1 (2 Discs) NTIS Order No. PB89-226666; "DARPA Resource Management Continuous Speech Database (RMI) Speaker-Independent Training Data," November 1989 NIST Speech Disc 2-3.1 (1 Disc) NTIS Order No. PB90-500539; "DARPA Extended Resource Management Continuous Speech Speaker-Dependent Corpus (RM2)," September 1990 NIST Speech Discs 3-1.2, 3-2.2 NTIS Order No. PB90-501776; "DARPA Acoustic-Phonetic Continuous Speech Corpus (TIMIT)," October 1990 NIST Speech Disc 1-1.1 NTIS Order No. PB91-0505065; "Studio Quality Speaker-Independent Connected-Digit Corpus (TIDIGITS)," NIST Speech Discs 4-1.1, 4-2.1, 4-3.1 NTIS Order No. PB91-505592), (for example 16 bit resolution collected at either 44,100 Hz, or 16,000 Hz) input at block 180 in FIG. 1.

The input speech data set 180 is first resampled to 8,000 KHz using the resampling program described in (A General Program to Perform Sampling Rate Conversion of Data by Rational Ratios," from "Programs for Digital Signal Processing," Ed.: Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Processing Society; IEEE Press, 1979; Section 8.2, pp. 8.2-1 to 8.2-7 by R. E. Crochiere) in function block 182 of FIG. 1. This is done after supplying it with a codec band-pass filter designed using a modified version of the MAXFLAT routine described in ("Design Subroutine (MAXFLAT) for Symmetric FIR Low Pass Digital Filters With Maximally-Flat Pass and Stop Bands" from "Programs for Digital Signal Processing," Ed.: Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Processing Society; IEEE Press, 1979; Section 5.3, pp. 5.3-1 to 5.3-6 by J. Kaiser) in function block 182 of FIG. 1. This filter is illustrated in FIGS. 2, 3, and 4. The passband characteristic of this filter is designed to closely approximate the coding/decoding, or "codec" filters used in contemporary U.S. telephonic equipment. The placement of the passband, 3 db points and the transition bandwidths are critical to the effectiveness of the invention. It is possible to design a codec filter for recognition training which will provide good recognition on local telephone lines, but not for long-distance lines. To avoid such a problem, these characteristics should be placed at say, 300 Hz for the lower 3 db point, and at 3,600 Hz for the upper 3 db point. The transition bandwidths should be 400 Hz, and 800 Hz respectively. This allows for a passband from 500 Hz, to 3,200 Hz. The passband ripple must be no more than 0.1 percent deviation from unity, throughout the pass band, to approximate the characteristics of actual codec filters.

It is important to note that the Sphinx recognition engine, and other recognition engines such as the Tangora, are sensitive to spectral distortions introduced by linear filters,

which do not have flat frequency response in the passband, since the primary speech recognition features are derived from frequency spectra, and their derivatives such as cepstra. Somewhat minor deviations from flat passband response have been shown in our laboratory to result in degradations of several percent in the absolute recognition error rates, for complex recognition tasks. Therefore a maximally flat design algorithm is required. The sensitivity of the Sphinx recognition engine to "spectral tilt" has been noted in ("Acoustical and Environmental Robustness in Automatic Speech Recognition," A. Acero; Carnegie Mellon University, Department of Electrical and Computer Engineering; April 1990; CMU-CS-88-148). Therefore, a MAX-FLAT, or comparably low passband ripple design is required.

The rate conversions required to resample to 8,000 KHz from 44,100 KHz are too demanding for the version of MAXFLAT provided in ("Design Subroutine (MAXFLAT) for Symmetric FIR Low Pass Digital Filters With Maximally-Flat Pass and Stop Bands" from "Programs for Digital Signal Processing," Ed. "Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Processing Society; IEEE Press, 1979; Section 5.3, pp. 5.3-1 to 5.3-6 by J. Kaiser) and it provides only for the design of low-pass filters, when a band-pass characteristic is required for the codec type filter required. The design characteristics for this routine are given by two parameters, beta, and gamma, representing the 3 db point and the transition bandwidth in normalized frequency, with the Nyquist frequency mapping to 0.5, and the sampling frequency mapping to 1.0. It is suggested, by Kaiser ("Design Subroutine (MAXFLAT) for Symmetric FIR Low Pass Digital Filters With Maximally-Flat Pass and Stop Bands" from "Programs for Digital Signal Processing," Ed. "Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Processing Society; IEEE Press, 1979; Section 5.3, pp. 5.3-1 to 5.3-6 by J. Kaiser), that gamma should be restricted to values "... not much smaller than 0.05." Values lower than this require the routine to be modified to increase the computing precision floating point numbers used, and for the filter coefficient buffers to be expanded from 200, to 4096, since the number of terms in such a filter are roughly proportional to the twice the inverse square of gamma. This accomplished, filters with gamma values as small as 0.005, or about ten times smaller 0.05, which are required for the 44,100 KHz to 8,000 KHz conversion were designed. Two low-pass filter designs, a low-pass to high-pass conversion, and convolutional combination of the high-pass, and low-pass were required to achieve the required band-pass characteristic.

With this filter design accomplished, the 44,100 Hz data was converted to 8,000 Hz in function block 182 of FIG. 1, using the resample algorithm described in ("A General Program to Perform Sampling Rate Conversion of Data by Rational Ratios," from "Programs for Digital Signal Processing," Ed.: Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Processing Society; IEEE Press, 1979; Section 8.2, pp. 8.5-1 to 8.2-7 by R. E. Crochiere), providing a codec passband characteristic, which very closely approximates the passband for U.S. long distance telephony equipment. This results in a 16-bit, low-noise signal, which must be treated according to the steps 2.) and 3.) below.

A similar band-pass characteristic, and rate reduction is required for the 16,000 Hz speech samples used in this training technique, with the exception that the transition band requirements are less demanding, and fewer filter weights are required to achieve the passband flatness char-

acteristic desired. FIGS. 2, 3, and 4 again show the Impulse, Magnitude, and Log Magnitude responses of the codec filter as it was implemented for the training one of the pre-training resampling operation.

2.) Scaling to Normalize the Dynamic Range

The speech sample utterances are then read individually, and scaled to a dynamic range of 14-bits, in function block 184 of FIG. 1.

3.) mu-law Companding

Each speech sample is then reduced from 16-bit precision to 8-bit precision using mu-law compression in function block 186 of FIG. 1, as is well known in the public literature, such as ("Digital Telephony and Network Integration," B. Keiser, E. Strange; Van Nostrand Reinhold Company Inc., 1985, pp. 26-31). The 8-bit compressed data is then expanded, also according to the mu, law formula back to 14-bits.

This results in simulated telephone channel speech data set in block 188 of FIG. 1, which has a quantization noise level which increases, and decreases with the signal strength, and maintains an approximately constant signal-to-noise ratio. This introduces the audible "crackle" noise present in telephony voice signals, particularly when the speaker is loud.

This treatment of speech utterance data 180, which may have been gathered at various bandwidths higher than telephone data, is used to train in block 190 of FIG. 1, a speech recognizer 50 of FIG. 1 for use over telephone equipment. Acoustic training 190 generates phoneme models in block 192 and quantization code books 105 in FIG. 1. This allows practical speech recognition at telephonic bandwidth using the Sphinx speech recognition engine.

RECOGNIZER TRAINING USING SIMULATED TELEPHONE CHANNEL DATA

Dual training sessions are then run so that two code book sets 105 and two phoneme model sets 192 are created, one for telephony, and one for high bandwidth. Each set of code books in 105 and each phoneme model set in 192 is kept separately, and run separately depending on the requirement of the user for high bandwidth, local recognition, or telephony bandwidth. At either bandwidth, the auto-correlation coefficients are extracted to derive Cepstrum coefficients. The Cepstrum coefficients are run through the vector quantizer 104 to classify which is its nearest neighbor for the frame. Thus each speech time-series frame is reduced to three bytes representing the frame, as described in ("Large Vocabulary Speaker and Dependent Continuous Speech Recognition: The Sphinx System," Kai-Fu Lee; Carnegie Mellon University, Department of Electrical and Computer Engineering; April 1988; CMU-CS-88-148).

The sets of quantized values are sent to the beam search process 106. The beam search 106 is a grammar guided Hidden Markov Model search process called a Viterbi beam search. This grammar guided search uses a word pair grammar to reduce the search space at any given point.

Another point of the invention is that the recognition system can process both local and long distance calls by placing the cutoff points of the run time data conditioning and rate conversion conversion filter to a bandwidth approximating the narrower of the two bandwidths, so that either type of call will correspond to the bandwidth used in the channel simulator described above. The 3 db point and transition band characteristics should closely approximate those of the upper transition band of the telephony codec filter used in the training and described above.

The beam search (block 106) matches time series derived in the vector quantization, to word sequences from within the word pair grammars, defining each context. The Recognition Server communicates with user applications or Recognition Clients (block 110). The invention's architecture can have multiple front end (workstations) communicating to a single back end or multiple front ends communicating to multiple back ends.

The system is organized and implemented for different levels of operation. For communication networks with a very high data rate, the speech samples could be communicated directly to the system executing the back-end, for front end data compression. A plurality of raw digital speech data streams could be sent to the server containing the back end for multiple users. For a telephony system, multiple channels go to one back end, or multiple users come in to the front end and back end together.

The system is primarily organized around the speech recognition functions deployed as speech recognition servers. The system is guided by any one of a plurality of word pair grammars the application has chosen as the current context. The application has interfaces to the speech recognition system with Application Program Interface (API) calls supporting functions like initializing procedures, status codes and commands ("IBM Continuous Speech Recognition System Programmers Guide," B. Booth, 1992, currently unpublished, available on request). The application will request a certain type of operation or ask the recognition server to load a certain recognition context and to activate the context for recognition when required. The tasks are pre-loaded by the server, usually when the application is first executed. They are then sequentially activated, as required by the activity of the application program.

A set of API calls in the recognition server (block 108) allows user applications (block 110) to request the services of the speech recognition system. user application programs (block 110) can be running on the same computer or a different computer as the various components of the recognition server. If it is on the same computer, the application program (block 110) might interface with the recognition server through shared memory and semaphores, supported by the operating system. If the application program (block 110) and recognition server are on a different computers, communication can be arranged via an RS232 interface, or Remote Procedure Calls (RPC). RPC being well known in the programming literature ("AIX Distributed Environments: NFS, NCS, RPC, DS Migration, LAN Maintenance and Everything," IBM International Technical Support Centers, Publication GG24-3489, May 8, 1990).

Typical examples of user applications may include: Executive Information Systems, Database Access via verbal query, software problem reporting systems, and so forth.

Another example is a telephone answering voice response unit (VRU) which could call on the recognition server to take advantage of the its services. We have implemented versions of these servers on the RISC System 6000[™], and PS/2[™] with OS/2[™].

The Direct Talk 6000[™] is a similar telephony VRU system. In Direct Talk 6000[™] rather than dealing with single telephone lines, the VRU system could require processing of a T1 line (with 24 conversation channels, possibly active simultaneously).

The recognition server architecture can handle multiple clients, as would be required to process such high-volume telephony applications as DirectTalk[™].

The user applications can pre-register many contexts: a

restaurants locator, a hard disk help desk, or a software help desk can all pre-register multiple contexts hierarchically. With each application, several users can be inputting speech streams. Each application will tell the recognition server to perform a recognition under a particular context for a particular speech stream, as appropriate for the task being executed.

In other words, multiple users dealing with the same API interface will register all their tasks, with one, or possibly several versions of the recognition server. The system arranges to avoid redundantly loading recognition tasks for multiple users, by checking if the requested task has already been loaded.

The task building (block 130) has several basic sources for its input. One is a U.S. English dictionary (block 132), which is a base dictionary with the pronunciations of twenty thousand words in it. The supplemental dictionary (block 138) is application specific, and allows for the addition of the pronunciation of words not found in the base dictionary. This would typically consist of proper nouns, acronyms, and the like, which a particular application requires for recognition.

The base U.S. dictionary (block 132) supplies words and the phoneme strings drawn on by the Task Builder (block 134). The Task Builder also draws on an appropriate task Baukus-Naur Form (BNF) grammar to determine what can be recognized by the speech server under the task, from the Task BNF library (block 136). For example, in an application which provides information on area restaurants, a first context may be the type of restaurant the caller wants, e.g., French, Italian, Chinese and a second the type was established would be the restaurants in that particular category. The task builder analyzes the BNF to find all the words that are required for the pattern matching and draws out the phoneme representation from the general U.S. dictionary (block 132). Inevitably, every particular application has its own sub-vocabulary which must be added to the system and these are stored in the supplemental dictionaries. For example, in a restaurant help desk, there are generic English words, such as: "Italian", "French", "Spanish", etc., which would be found in the standard U.S. dictionary. However, restaurant names, particularly in foreign languages, e.g., "Cherchez Les Femmes", "Chateau Voulez", but also unusual names for an American restaurant, e.g., J. J. Muldoon's, will not be in any normal dictionary, and must be added to the task supplemental dictionary (block 138). These supplemental dictionaries (block 138) can also contain local vocabulary that is in the base General English (block 132) dictionary which override the pronunciations.

The task builder (block 134) analyzes the input BNF grammar, and generates a grammar which is a list of each word in the grammar and a sub-list of all the words that can follow. Thus each word in the grammar has a list attached to it of legal following words and a pointer to the phoneme representation of each word, (called phoneme models 192 in FIG. 1). The phoneme models 192 are Hidden Markov Models of observing the various VQ values. The hidden Markov models are a group of discrete probability distributions, for the VQ values (as in block 104). These provide the probability of the occurrence of VQ values, given that the hidden Markov state machine is in a particular state within a phoneme. The public literature contains excellent descriptions of Hidden Markov Models in ("A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," L. Rabiner; Readings in Speech Recognition; Ed.: A. Waibel, K. Lee; Morgan Kaufmann; 1990; pp. 267-296), as well as elsewhere.

The Beam Search (block 106) uses word models made of concatenated HMM phoneme models 192 from a large table of context sensitive triphones which are generated during the training process. These are used to make an optimal estimate of the word sequence which best explains the observed sequence of VQ values. The beam searcher (block 106) uses the word grammars to select the phoneme models 192 from which to construct the word models used in the search.

The user applications control the recognition server. For example, DirectTalk/2[™], an IBM Program Product described in ("IBM CallPath DirectTalk/2 General Information and Planning Manual," International Business Machines Publication No. GB35-4403-0; 1991), could be a user application; it is able to answer the phone and perform restaurant locator functions. The restaurant locator application would use the DirectTalk/2[™] system to indicate to the recognition server that it has sixteen contexts and issue a request to pre-load the contexts which are part of the Restaurant Locator help desk. As the application progresses, it requests the context switching of the recognition server. A user calls via the telephone for telephone help. The restaurant locator then requests the recognition server to perform a voice recognition under the first level context. Control and data are exchanged over the API between the recognition server, and the user application. Multiple instances of the DirectTalk/2[™] system could use the same recognition server.

The speech recognition server acquires speech data until a (user adjustable, but most commonly 0.6 seconds) period of silence. Recognition is terminated when this period is observed, and it is assumed that the person is done speaking.

The speech recognition system described herein, is architected to allow multiple deployments, on multiple hardware platforms, and multiple software configurations. For example, one possible architecture is shown in FIG. 5, which provides a physical mapping of the logical architecture 50, discussed above, onto a physical implementation of workstations connected via a local area network 160. Each workstation 150, 150', 150" in this architecture can run multiple independent user applications, and each is master to the recognition server 50 as a slave processor. The PBX 170 is connected to outside telephone lines and delivers a telephony bandwidth data stream to the analog digital conversion 100 of the recognition server 50, also shown in FIG. 1. The text representing recognized speech is returned from the recognition server to the user applications in workstations 150, 150', 150".

TRAINING PROCESS

The training procedure uses a large library of known utterances and their textual transcripts, to estimate the parameters of the phonemes HMMs 192 used in pattern matching of word models to text in the beam search process.

First, the transcripts are used to retrieve the phonemes, representing the pronunciation of the words in the training set, from the General English dictionary.

Next the parameters of phoneme HMMs 192 are estimated in the context of preceding, and following phonemes, (called triphones) to provide for effective estimation of coarticulation effects. The estimation procedure used is the Baum-Welch Forward/backward iteration algorithm described in ("A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," L. Rabiner; Readings in Speech Recognition; Ed.: A. Waibel, K. Lee; Morgan Kaufmann; 1990; pp. 267-296). The parameters of

the HMMs are iteratively adjusted so as to maximize the probability that the trained triphone HMMs would have generated the time series of VQ values observed in the training set.

There are many parameters for every hidden Markov phoneme model, there being 7 states and 12 transition arcs in each hidden state machine. Associated with each transition arc are associated 256 discrete elements in the probability distribution for each of the three code books. The triphone HMMs parameters that result from the training procedure are clustered sharply to reduce the number of triphones required to adequately represent the coarticulation effects present in continuous speech.

Training is performed with a combination of low bandwidth speech which is gathered via the local telephone exchange and high bandwidth speech from a microphone. The high bandwidth speech is processed by the telephone channel simulator 185 described herein, in accordance with the invention. All three code books are compiled at this stage. The code book versions include Cepstrum, differential Cepstrum, Power and differential Power as described in ("Large Vocabulary Speaker and Dependent Continuous Speech Recognition: The Sphinx System," Kai-Fu Lee; Carnegie Mellon University, Department of Electrical and Computer Engineering; April 1988; CMU-CS-88-148).

Each of these three code books stored in quantization code books 105 and are used in the run time Vector Quantization process. The effect of the telephone network is simulated here by data pre-conditioning so that the statistical properties of these feature code books will be adjusted in the same manner in which the public telephony network will adjust them. This procedure has resulted in a substantial increase in accuracy in actual telephonic speech recognition with calls originating from a variety of locations in the continental U.S.

Reference can be made to the flow diagram of FIG. 6 which describes the telephony channel simulation process 200 for training a speech recognizer 50 to respond to speech obtained from telephone systems, for example, through a PBX 170. The flow diagram of FIG. 6 represents a computer program method which can be executed on the data processor 50 of FIG. 5.

The process 200 starts with step 202 in which a speech data set is input to a speech recognition training processor 50, whose bandwidth is higher than telephone bandwidth. Example high bandwidth speech data sets are identified in references ("Speech Corpora Produced on CD-ROM Media by The National Institute of Standards and Technology (NIST)," April 1991; "DARPA Resource Management Continuous Speech Database (PMI) Speaker Dependent Training Data," September 1989 NIST Speech Discs 2-1.1, 2-2.1 (2 Discs) NTIS Order No. PB89-226666; "DARPA Resource Management Continuous Speech Database (RMI) Speaker-Independent Training Data," November 1989 NIST Speech Disc 2.3.1 (1 Disc) NTIS Order No. PB90-500539; "DARPA Extended Resource Management continuous Speech Speaker-Dependent Corpus (RM2)," September 1990 NIST Speech Discs 3-1.2, 3-2.2 NTIS Order No. PB90-501776; "DARPA Acoustic-Phonetic Continuous Speech Corpus (TIMIT)," October 1990 NIST Speech Disc 1-1.1 NTIS Order No. PB91-0505065; "Studio Quality Speaker-Independent Connected-Digit Corpus (TIDIG-ITS)," NIST Speech Discs 4-1.1, 4-2.1, 4-3.1 NTIS Order No. PB91-505592). This corresponds to data input block 180 in FIG. 1.

Then, in step 204 in FIG. 6, the speech data set is

decimated to obtain a decimated speech data set having the telephone bandwidth. This corresponds to function block 182 of FIG. 1. The decimated speech data set may have bandwidth which is any bandwidth lower than the higher bandwidth of the input speech data set. Decimation process are described in reference (A General Program to Perform Sampling Rate Conversion of Data by Rational Ratios;" from "Programs for Digital Signal Processing," Ed.: Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signals Processing Society; IEEE Press, 1979; Section 8.2, pp. 8.2-1 to 8.2-7 by R. E. Crochiere).

Then, in step 206 in FIG. 6, applies a bandpass digital filter to the decimated speech data set, which characterizes transmission characteristics of telephone equipment. This corresponds to function block 182 of FIG. 1. This is done to obtain a filtered speech data set. The bandpass digital filter should have a maximally flat design algorithm.

Then, in step 208 in FIG. 6, the amplitude of the filtered speech data set is rescaled so that its maximum dynamic range matches the maximum range of uncompanied telephone speech. This corresponds to function block 184 of FIG. 1. This is done to obtain a rescaled speech data set. The rescaling step can result in the maximum dynamic range matching the maximum dynamic range of uncompanied mu-law telephone speech. Alternately, the rescaling step can result in the maximum dynamic range matching the maximum dynamic range of uncompanied A-law telephone speech.

Then, in step 210 in FIG. 6, modifies the rescaled speech data set, with quantization noise representing a sequence of companding and uncomparing a speech signal in a telephone system. This corresponds to function block 186 of FIG. 1. This is done to obtain a modified speech data set. The modifying step can have quantization noise as mu-law noise. Alternately, the modifying step can have quantization noise as A-law noise.

Then, step 212 in FIG. 6, inputs the modified speech data set into the speech recognition processor 50, to train statistical pattern matching data units. This corresponds to output data block 188 of FIG. 1. The simulated telephone channel speech 185 is then used by the acoustic training process 190 to generate phoneme models 192 characteristic of telephone code books 105 characteristic of telephone speech.

Then, in step 214 in FIG. 6, speech recognition can be performed using the speech recognition processor 50, on voice signals from a telephone system, such as, signals from the PBX 170 of FIG. 5.

It should be noted that the conversion of high bandwidth speech using the telephony channel simulator (block 185) is not limited to continuous speech recognizers, but applies to a variety of speech recognition processors, such as, the IBM TangoraDictation System and Dragon Systems, Newton Mass., Dragon 30k Dictate and Kurzweil Applied Intelligence, Voice Report, Waltham, Mass. and others described in ("The Spoken Word," Kai-Fu Lee, et al., Byte Magazine,

July 1990, Vol. 15, No. 7; pp. 225-232).

While the invention has been described with reference to a preferred embodiment, it will be understood by those skilled in the art that various changes can be made to the architecture without departing from the spirit and scope of the invention. Accordingly, the invention shall be limited only as specified in the following claims.

We claim:

1. A method for training a speech recognition processor to respond to speech obtained from telephone systems, comprising the steps of:

inputting a speech data set to a speech recognition training processor, said data set having a bandwidth higher than a telephone bandwidth;

decimating said inputted speech data set in said training processor to obtain a decimated speech data set having said telephone bandwidth;

applying a bandpass digital filter to said decimated speech data set in said training processor, said filter characterizing transmission characteristics of telephone equipment, for obtaining a filtered speech data set;

rescaling the amplitude of said filtered speech data set in said training processor, so that the maximum dynamic range of said filtered speech data set matches the maximum dynamic range of uncompanied telephone speech, to obtain a rescaled speech data set;

modifying said rescaled speech data set in said training processor, with quantization noise representing companding and uncomparing a speech signal in a telephone system, to obtain a modified speech data set;

inputting said modified speech data set into a hidden Markov model speech recognition processor to train statistical pattern matching data units;

performing speech recognition on voice signals from a telephone system with said speech recognition processor.

2. The method of claim 1 wherein:

said telephone bandwidth is any bandwidth lower than said higher bandwidth.

3. The method of claim 1 which further comprises:

said bandpass digital filter has a maximally flat design algorithm.

4. The method of claim 1 wherein said rescaling step results in a maximum dynamic range matching a maximum dynamic range of uncompanied mu-law telephone speech.

5. The method of claim 1 wherein said rescaling step results in a maximum dynamic range matching a maximum dynamic range of uncompanied A-law telephone speech.

6. The method of claim 1 wherein said modifying step has quantization noise as mu-law noise.

7. The method of claim 1 wherein said modifying step has quantization noise as A-law noise.

* * * * *

[54] **SYSTEM FOR MULTILEVEL SECURE DATABASE MANAGEMENT USING A KNOWLEDGE BASE WITH RELEASE-BASED AND OTHER SECURITY CONSTRAINTS FOR QUERY, RESPONSE AND UPDATE MODIFICATION**

[76] Inventors: **Bhavani M. Thuraisingham**, 209 Katardin Rd., Lexington, Mass. 02173; **William R. B. Ford**, 13 Coach Rd., Billerica, Mass. 01862; **Marie S. Collins**, 4452 Wordsworth Rd., Plano, Tex. 75093; **Jonathan P. O'Keeffe**, 70 JFK Blvd., Somerset, N.J. 08873

[21] Appl. No.: 767,258

[22] Filed: Sep. 27, 1991

[51] Int. Cl.⁵ G06F 15/40

[52] U.S. Cl. 395/600; 380/4; 380/25; 395/650; 395/725; 364/DIG. 1; 364/274; 364/282.1; 364/283.3; 364/283.4; 364/286.5; 364/286.6

[58] Field of Search 395/51, 600, 650, 725; 380/4, 25; 364/578

[56] References Cited

U.S. PATENT DOCUMENTS

4,123,747	10/1978	Lancto et al.	380/25
4,672,572	6/1987	Alsberg	380/23
4,791,561	12/1988	Huber	395/600
4,866,635	9/1989	Kahn et al.	395/51
4,882,752	11/1989	Lindman et al.	380/25
5,016,204	1/1991	Simoudis et al.	364/578

5,075,884	12/1991	Sherman et al.	395/650
5,126,728	6/1992	Hall	380/825.3
5,129,685	10/1992	Kung	395/575
5,175,800	12/1992	Galis et al.	395/51
5,193,185	3/1993	Lanter	395/600
5,201,047	4/1993	Waki et al.	395/600
5,265,221	11/1993	Miller	395/725
5,278,946	1/1994	Shimada et al.	395/62

OTHER PUBLICATIONS

Lunt et al, The SeaView Security Model, IEEE Trans. on Software Engineering, vol. 16, No. 6, 1990, pp. 593-607.

Lunt et al, A Near α Term Design for the SeaView Multilevel Database System, Proc. of the IEEE Symposium on Security and Privacy, 1988, pp. 234-244.

Primary Examiner—Thomas C. Lee

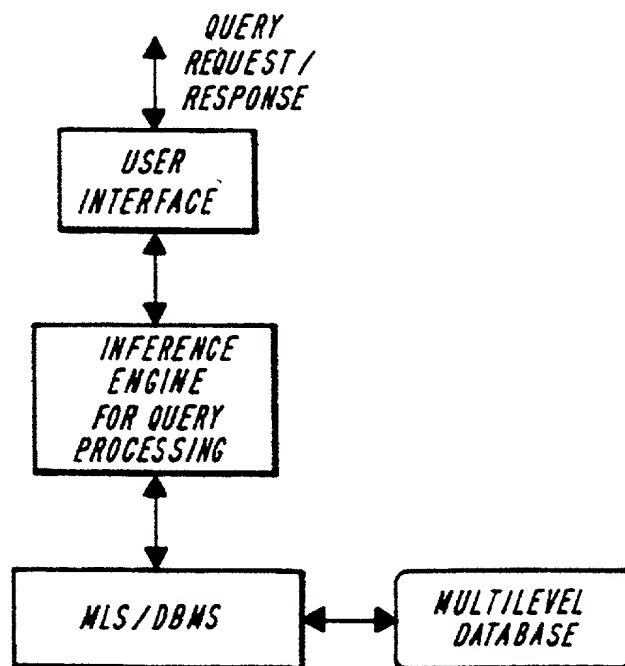
Assistant Examiner—Wayne Amsbury

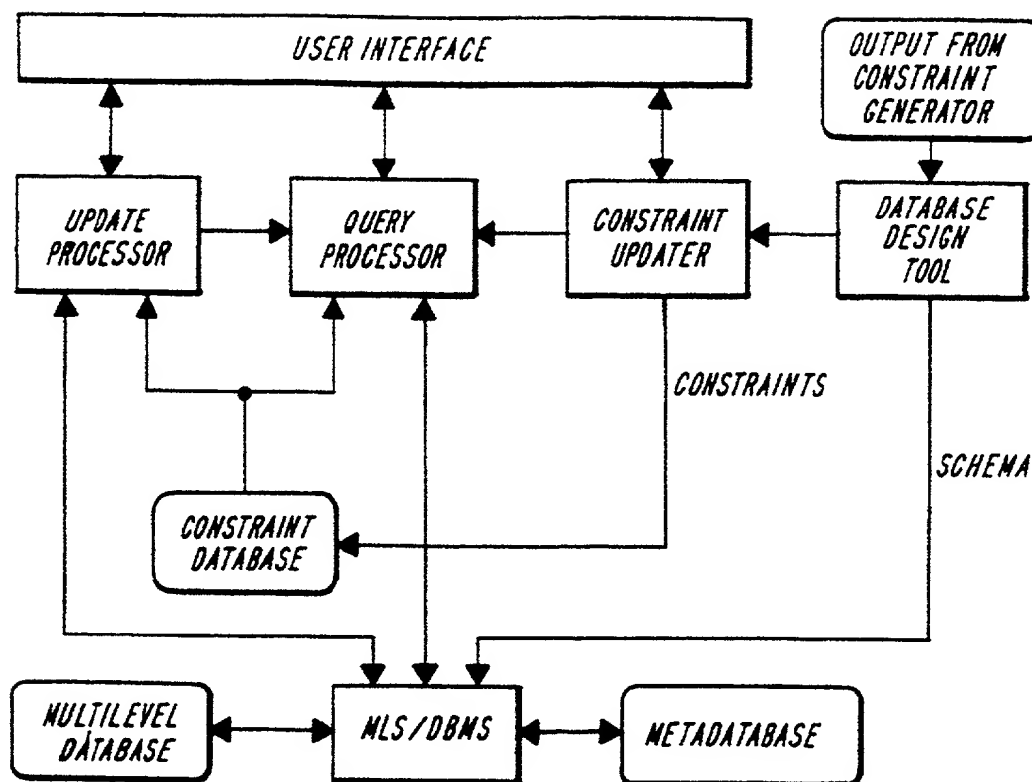
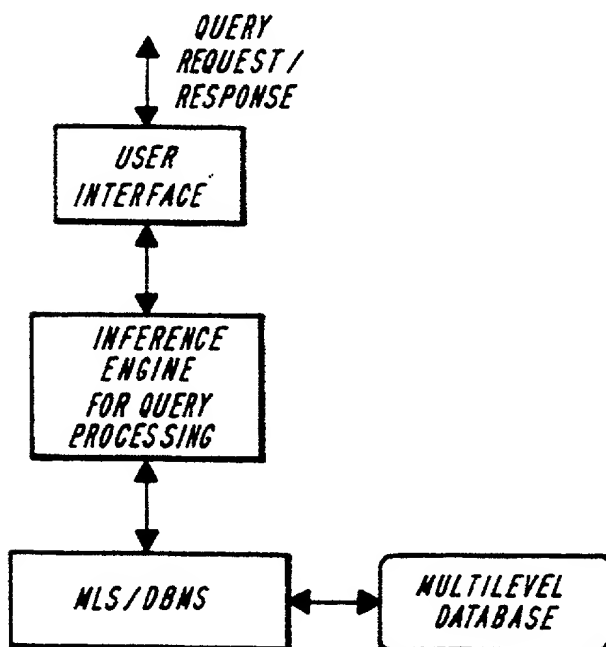
Attorney, Agent, or Firm—Choate, Hall & Stewart

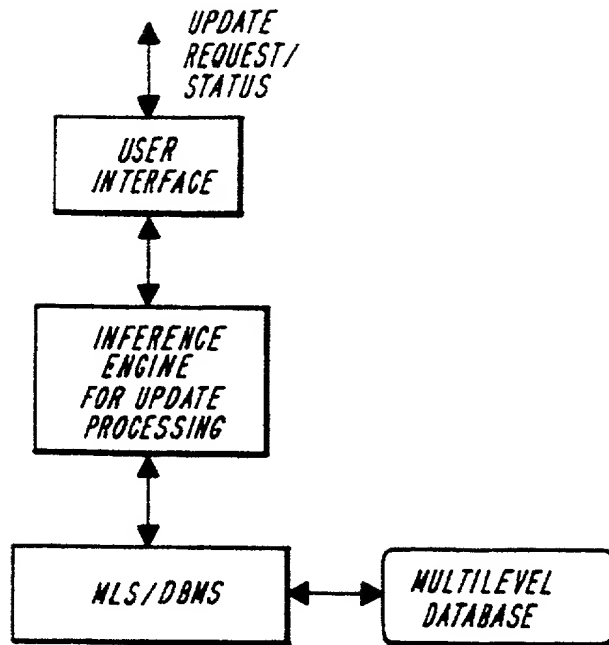
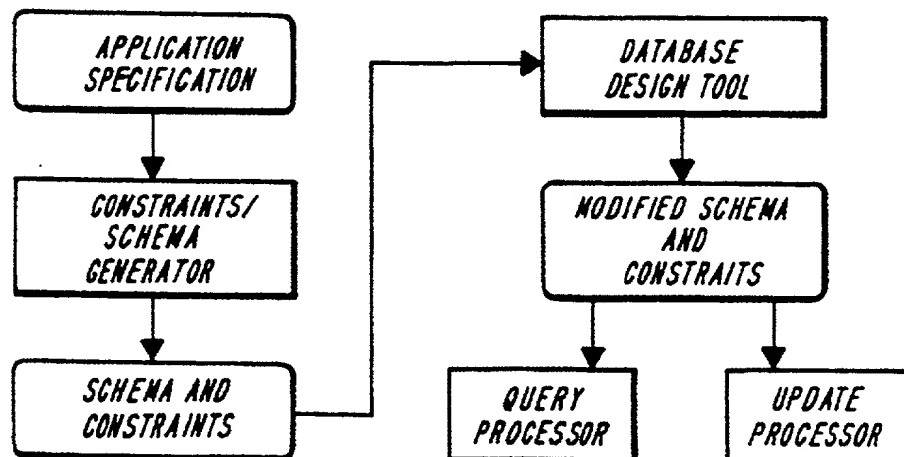
[57] ABSTRACT

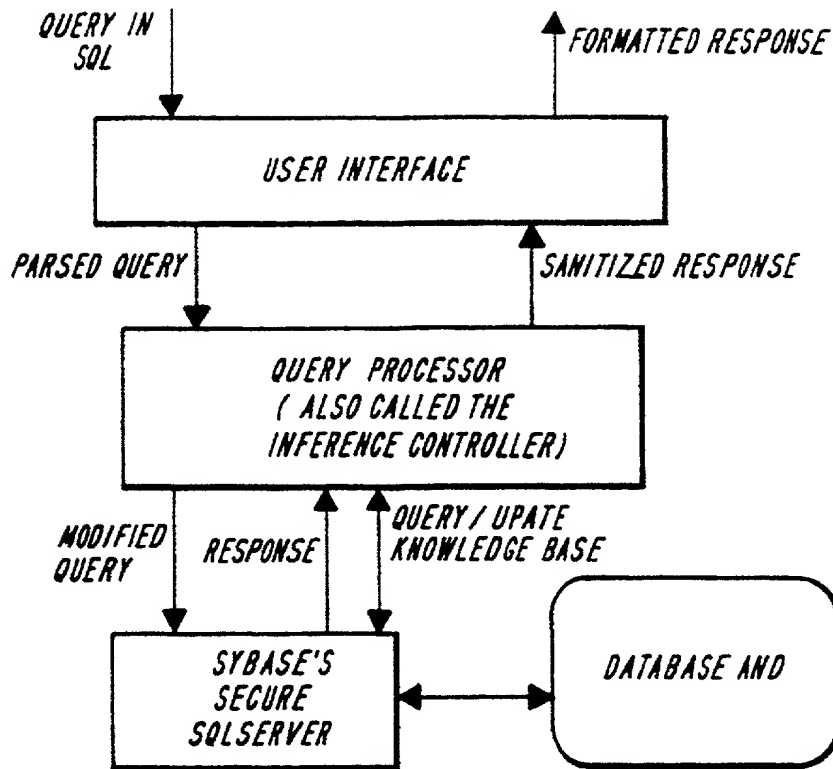
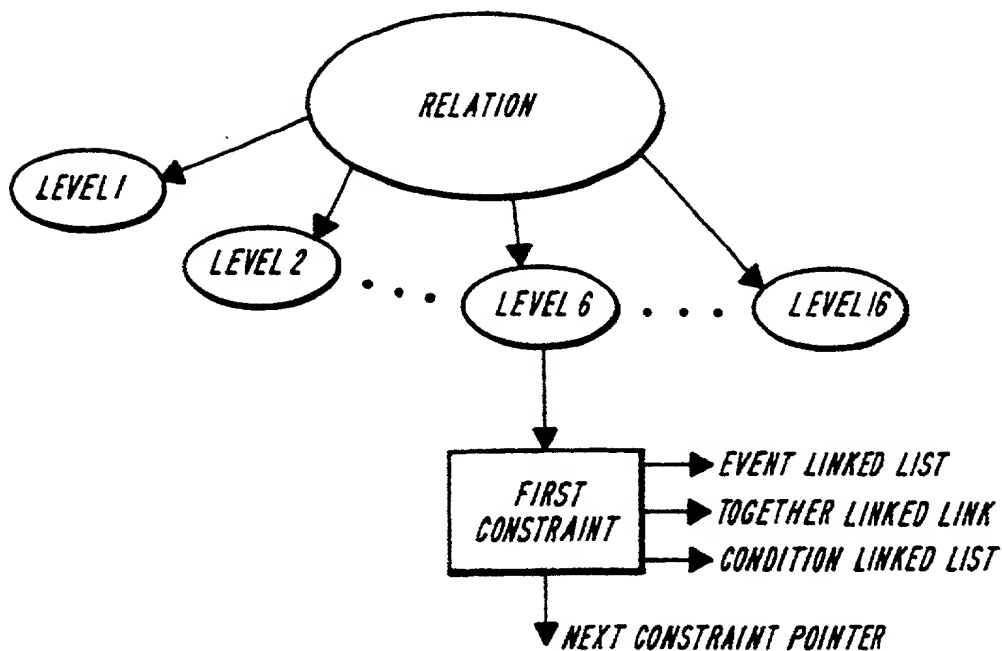
Apparatus for an integrated architecture for an extended multilevel secure database management system. The multilevel secure database management system processes security constraints to control certain unauthorized inferences through logical deduction upon queries by users and is implemented when the database is queried through the database management system, when the database is updated through the database management system, and when the database is designed using a database design tool.

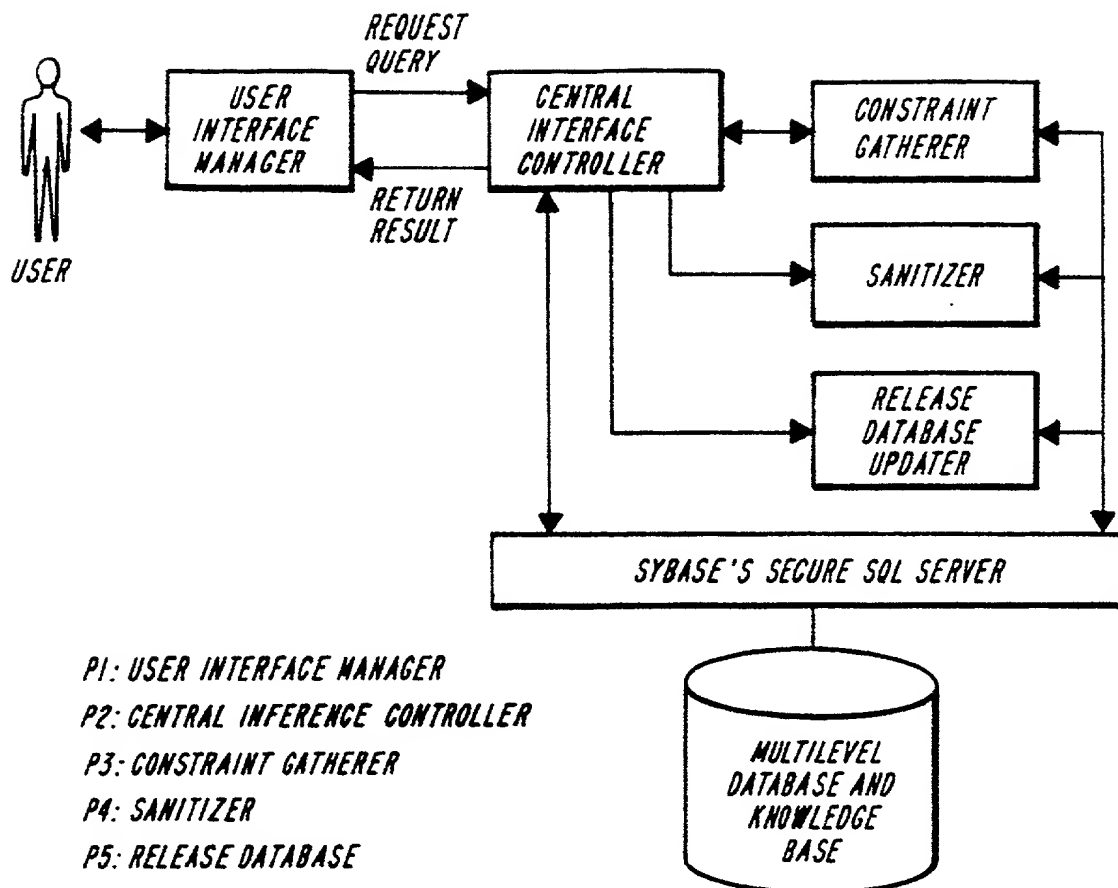
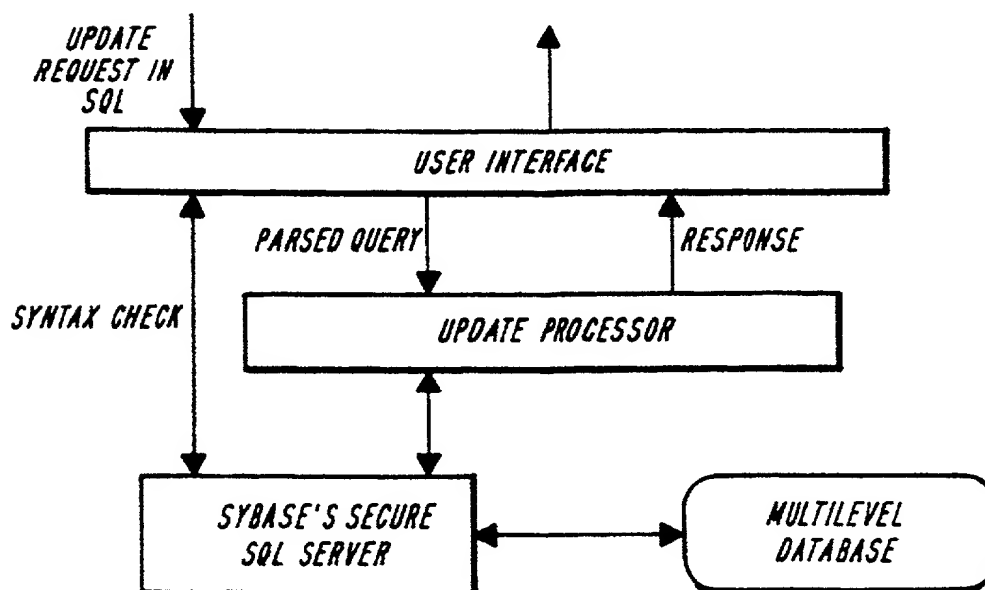
4 Claims, 5 Drawing Sheets

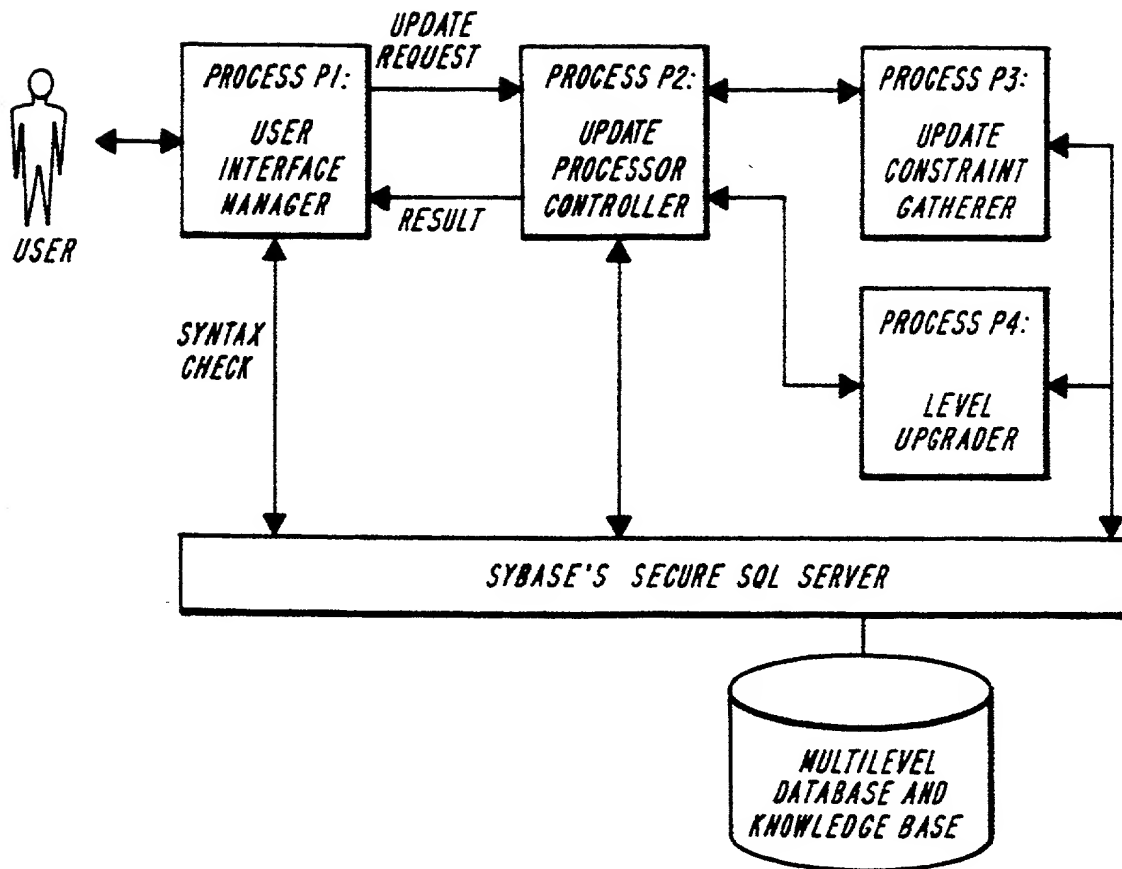


**FIG. 1****FIG. 2**

*FIG. 3**FIG. 4**FIG. 5*

**FIG. 6****FIG. 7**

**FIG. 8****FIG. 9**

**FIG. 10**

SYSTEM FOR MULTILEVEL SECURE DATABASE MANAGEMENT USING A KNOWLEDGE BASE WITH RELEASE-BASED AND OTHER SECURITY CONSTRAINTS FOR QUERY, RESPONSE AND UPDATE MODIFICATION

BACKGROUND OF THE INVENTION

It is possible for the users of a database management system to draw inferences from the information that they obtain from the database. The inference process can be harmful if the inferred knowledge is something that the user is not authorized to acquire. That is, a user acquiring information which he is not authorized to know has come to be known as the inference problem in database security. In a multilevel operating environment, the users are cleared at different security levels as they access a multilevel database where the data is classified at different sensitivity levels. A multilevel secure database management system (MLS/DBMS) manages a multilevel database where its users cannot access data to which they are not authorized. Currently available multilevel secure database management systems cannot provide a solution to the inference problem, where users of the system issue multiple requests and consequently infer unauthorized knowledge.

Two distinct approaches to handling the inference problem have been proposed in the past. They are:

- (i) Handling of inferences during database design.
- (ii) Handling of inferences during query processing.

The work reported in Morgenstern, M., May 1987, "Security and Inference in Multilevel Database and Knowledge Base Systems," Proceedings of the ACM SIGMOD Conference, San Francisco, Calif.; Hinke, T., April 1988, "Inference Aggregation Detection in Database Management Systems," Proceedings of the IEEE Symposium on Security and Privacy, Oakland, Calif.; Smith, G., May 1990, "Modelling Security-Relevant Data Semantics," Proceedings of the 1990 IEEE Symposium on Security and Privacy, Oakland, Calif.; and Lunt, T., May 1989, "Inference and Aggregation, Facts and Fallacies," Proceedings of the IEEE Symposium on Security and Privacy, Oakland, Calif. focuses on handling inferences during database design where suggestions for database design tools are given.

In contrast, the work reported in Thuraisingham, B., December 1987, "Security Checking in Relational Database Management Systems Augmented with Inference Engines," Computers and Security, Volume 6, No. 6.; Thuraisingham, B., August 1990, The Use of Conceptual Structures in Handling the Inference Problem, Technical Report M90-55, The MITRE Corporation, Bedford, Mass.; Keefe, T., B. Thuraisingham, and W. Tsai, March 1989, "Secure Query Processing Strategies," IEEE Computer, Volume 22, No. 3, pp. 63-70 focuses on handling inferences during query processing.

Other work on handling the inference problem can be found in Buczkowski, L.J., and Perry, E.L., "Database Inference Controller," Interim Technical Report, Ford Aerospace Corporation, February 1989, where an expert system tool which could be used by the System Security Officer off-line to detect and correct logical inferences is proposed. Rowe, N., February 1989, "Inference Security Analysis Using Resolution Theorem Proving," Proceedings of the 5th International Conference on Data Engineering, Los Angeles, Calif. investigates the use of Prolog for handling inferences.

In Thuraisingham, B., August 1990, The Use of Conceptual Structures in Handling the Inference Problem, Technical Report M90-55, The MITRE Corporation, Bedford, Mass. various strategies that users could utilize to draw inferences are identified. This set of strategies is more complete than the one proposed in Denning, D.E., et al., "Views as a Mechanism for Classification in Multilevel Secure Database Management Systems," Proceedings of the IEEE Symposium on Security and Privacy, Oakland, Calif. 1986. In Thuraisingham, B., August 1990, The Use of Conceptual Structures in Handling the Inference Problem, Technical Report M90-55, The MITRE Corporation, Bedford, Mass., some preliminary ideas on novel approaches to handling the inference problem are discussed. These include approaches based on mathematical programming, inductive inference, information theory and game theory. Further, in Thuraisingham, B., August 1990, The Use of Conceptual Structures in Handling the Inference Problem. Technical Report M90-55, The MITRE Corporation, Bedford, Mass. complexity of the inference problem is analyzed based on concepts in recursive function theory.

The present application discloses an apparatus and method for designing a multilevel secure database management system that can resolve the inference problem via the effective use of security constraints. In the new system, some security constraints are handled during the query operation, some during the update operation, some during the database design operation. The major advance achieved by the invention disclosed herein over prior art is the use of security constraints in a novel way to handle the inference problem. In addition, prototypes which effectively handle these constraints are also disclosed. Further advances relate to the use of conceptual structures for representing and reasoning about multilevel applications, the development of a logic for secure data/knowledge base management systems and the development of a knowledge base inference controller.

SUMMARY OF THE INVENTION

The invention disclosed herein is an apparatus and method for querying, designing and updating a multilevel secure database management system in order to resolve the inference problem.

A method for processing security constraints in a multilevel secure database management system are disclosed. Security constraints are rules that assign security levels to data. This method is based on specifying security constraints as horn clauses. The apparatus and method disclosed here uses nine types of security constraints:

1. Simple constraints that classify a database, a relation or an attribute;
2. Content-based constraints that classify any part of the database depending on the value of some data;
3. Event-based constraints that classify any part of the database depending on the occurrence of some real-world event;
4. Association-based constraints that classify associations between attributes and relations;
5. Release-based constraints that classify any part of the database depending on the information that has been previously released;
6. Aggregate constraints that classify collections of data;

GetFindName makes the identification by matching the caller's keyed input (SMSI caller id(00), digit name, extension number) against a list of user ids generated from the Host. When the extension number or digit name is entered, a minimum number of keyed inputs are received and accepted by the Channel Process for identification. The Channel Process uses the keyed inputs to retrieve a list of user ids from the Host. Then it searches the list of user ids for a unique match.

When GetFindName has identified the caller or destination, it requests the user profile stored in the data base. For example, at log on to system services, the caller has the option of entering a keyed input of pound (#), his extension number, or his digit name. If the caller enters a keyed input of pound (#), GetFindName uses the SMSI caller id and if the caller enters his name or extension GetFindName uses the keyed input for identification.

In any case, when the keyed input is entered, GetFindName uses the list provided by the Host to identify the extension number or digit name. When a caller enters an extension or digit name as the destination when sending a message to another party, GetFindName uses the list provided by the Host in order to identify the receiver so that the message is sent to the correct destination.

15 PARAMETERS

CALLER'S USER ID. This parameter means that GetFindName is used to identify a caller.

DESTINATION USER ID. This parameter means that GetFindName is used to identify a receiver.

EDGES

- 0 = Found
- 20 1 = Not found
- 2 = Input incomplete. A time out has occurred.
- T1 = No input. A time out has occurred.
- T2 = Last time out has occurred
- HUP = The caller has hung up.

25

GetFindPassword

The GetFindPassword action compares the keyed input password with the password defined in the user profile. The last key that is entered must be a # key.

30 **PARAMETERS NONE REQUIRED.**

EDGES

- 0 = Password correct
- 1 = Password false
- 2 = Input incomplete (time out)
- 35 * = Mistake keyed. Stops the process in the event of a mistake.
- T1 = Time out
- T2 = Last repeat time
- HUP = The caller has hung up.

40 **EvaluateData**

This action is used to test the values of system variables with other variables or constants. The flow of the state table can then be altered based on the results of the evaluation.

PARAMETERS

45 **PARM 1:** Variable id or constant to test

PARM 2: Variable id or constant

EDGES

- 0 = Parm 1 less than Parm 2
- 1 = Parm 1 equal to Parm 2
- 50 2 = Parm 1 greater than Parm 2
- HUP = The caller has hung up.

AssignData

55 This action can be used in the state table to perform simple arithmetic or string concatenation. It can be used to preset variables to specific values before using them in an action, or as counters in the state table.

For example, the AssignData action can be used to do loop processing. If there are three prompts for a password, then after the first prompt, AssignData is used to loop back to the first prompt. The AssignData action

can also be used to pre-assign a variable to a given value before calling another action.

PARAMETERS

PARM 1: Operation (1 add; 2 Subtract; 3 Multiply, 4 Divide; 5 Assign only.

PARM 2: Variable id of buffer to contain the results

5 **PARM 3:** Variable id or constant for the first operand; if Param 1 is not 5, then also use this.

PARM 4: Variable id or constant.

EDGES

0 = Assignment complete;

1 = Assignment overflowed buffer;

10 HUP = The caller has hung up.

PlayVoice

15 Like the PlayPrompt action, this action plays digitized data on the voice channel. It is used to play voice segments, audio names, user greetings, or user messages (voice mail) from either the Host data base or the RecordVoice workspace area. This action is used after recording voice to allow the user to verify what he has recorded before saving it; or, in the case of user messages, it is used before sending the messages to the destination mailboxes. Each user message is assigned an active message number. This is the pointer, which is activated by current message header that is under examination.

20 PARAMETERS

PARM 1: VOICE TYPE

1 Voice segment (PlayVoice is used only by administrator to record or play voice segments.)

2 Audio name

3 User greeting

25 4 User message (PlayVoice uses the variable active message number specified by GetKey.).

The following parameters are required depending on the voice type:

VOICE SEGMENT

Param 2: Numeric buffer name containing the voice segment id

Param 3: Numeric buffer name containing the language code.

30 AUDIO NAME

Param 2: Character buffer name containing the user id.

USER GREETING

Param 2: Character buffer name containing the user id

Param 3: Numeric buffer name containing the greeting entry number.

35 **NOTE:** For all voice types, Param 2 can provide the workspace area. The workspace area is where the user can play the voice data that has been recently recorded, before making a decision to record, save, or delete the voice data.

EDGES

0 = Play completed

40 1 = Voice channel problem

2 = Voice record not found

HUP = The caller has hung up.

RecordVoice

45

This action is used to record voice as digitized voice data on the system disk into a voice segment, audio name, user greeting, or user message.

The voice data is first recorded into a temporary workspace area from which it can be replayed and verified with the PlayVoice action before storing it at its final destination through SaveVoice.

50 PARAMETERS

PARM 1: VOICE TYPE

1 Voice segment (RecordVoice is used only by the system administrator to record or play voice segments.)

2 Audio name

3 User greeting

55 4 User message (RecordVoice uses the variable active message number specified by GetKey.)

EDGES

0 = Recording completed

1 = Voice channel problem

- 2 = No voice recorded
 3 = Disk full
 T1 = Time out: xx seconds remaining to record before maximum time is reached. It is specified by the System Administrator.
 5 T2 = The maximum recording time has been reached. HUP = The caller has hung up.

SaveVoice

10 This action saves previously recorded voice data for the specified voice type. When recording voice, the data is always recorded into a temporary workspace first. This action copies the voice data from the workspace area to its destination (for example, voice segment id, audio name, or user greeting).

PARAMETERS**PARM 1: VOICE TYPE**

- 1 Voice segment (SaveVoice is used only by the system administrator to record or play voice segments.)
 15 2 Audio name
 3 User greeting
 4 User message (SaveVoice uses the variable active message number specified by GetKey.).

The following parameters are required depending on the voice type:

VOICE SEGMENT

- 20 **Parm 2:** Numeric buffer name containing the voice segment id
Parm 3: Numeric buffer name containing the language code.

AUDIO NAME

Parm 2: Character buffer name containing the user id.

USER GREETING

- 25 **Parm 2:** Character buffer name containing the user id
Parm 3: Numeric buffer name containing the greeting entry number.

USER MESSAGE

Parm 2: Numeric buffer name containing the receiver id
Parm 3: Mailbox id.

EDGES

- 30 0 = Save successful
 1 = Save unsuccessful
 HUP = The caller has hung up.

35 DeleteVoice

This action deletes previously recorded voice data for the specified voice type.

PARAMETERS**PARM 1: VOICE TYPE**

- 40 1 Voice segment (DeleteVoice is used only by the system administrator to record or play voice segments.)
 2 Audio name
 3 User greeting
 4 User message (DeleteVoice uses the variable active message number specified by GetKey.).

The following parameters are required depending on the voice type:

45 VOICE SEGMENT

Parm 2: Numeric buffer name containing the voice segment id
Parm 3: Numeric buffer name containing the language code.

AUDIO NAME

Parm 2: Character buffer name

50 USER GREETING

Parm 2: Character buffer name containing the user id
Parm 3: Numeric buffer name containing the greeting entry number.

NOTE: For all voice types, Parm 2 provides the workspace area.

EDGES

- 55 0 = Delete successful
 1 = Delete unsuccessful
 2 = Voice record not found
 HUP = The caller has hung up.

CheckStorage

This action is used to check system resources in order to allow an alternate flow through an application based on the resources available. It is normally used at the beginning of an application to determine if there are any storage problems. It is also used before recording to determine if there is space available and whether or not the item already exists.

PARAMETERS**Calling parameters**

PARM 1: Resource or item conditions to check:

- 1 Voice segment exists
- 2 Audio name exists
- 3 User greeting exists
- 4 Mailbox space is available
- 5 System disk storage space is available

Based on the condition specified above, the following parameters are also needed.

VOICE SEGMENT

Parm 2: Numeric buffer name containing the voice segment id

Parm 3: Numeric buffer name containing the language code

AUDIO NAME

Parm 2: Character buffer name containing the user id

Parm 3: Numeric buffer name containing the language code **USER GREETING**

Parm 2: Character buffer name containing a user id

Parm 3: Numeric buffer name containing the greeting entry number in the application profile.

MAILBOX SPACE

Parm 2: Character buffer id containing a user id

Parm 3: 1 Check space for new messages.
2 Check space for saved msgs.

EDGES

0 = Condition is true

1 = Condition is false

HUP = The caller has hung up

CheckMailbox

The CheckMailbox action checks the mailbox of the specified user id for incoming or outgoing mail. For example, if the message headers for the messages that are stored in the data base contain the sender's user id, date and time the message was sent, and message attributes such as message type and status. The first time the user invokes CheckMailbox, the active message number acts as a pointer to the current message header that is under examination. If the user continues to invoke CheckMailbox, the active message number acts as a pointer to subsequent message headers in the data base. In effect, when, the Check first entry and Check next entry parameters that are defined in the state table are invoked, the most recent message is played first, followed by any older messages.

PARAMETERS**PARM 1:**

- 1 Check first entry
- 2 Check next entry

PARM 2: (Parm 1 = 1)

- 1 Incoming new mail
- 2 Incoming old mail
- 3 Outgoing new mail

0 = No messages

1 = Messages retrieved

2 = Mailbox is busy

HUP = The caller has hung up.

UpdateMailbox

This action updates the message header entries in a message in order to discard or save a received mes-

sage, to send messages to other user id's, or to update the message's attributes. For example, with this action, the user can alter the selection type of a given message (for example, regular, urgent, or emergency), change the security level of a given message, or update the receiver id.

PARAMETERS

5 **PARM 1:** The attribute field to update containing the data to update the field.

EDGES

0 = Update complete

1 = Update failed

HUP = The caller has hung up.

10

UpdateUserProfile

This action is used to modify the user profile. Some fields of the user profile can be modified by the subscriber, such as password and selected language; while some fields cannot, such as name and number of messages. UpdateUserProfile allows the selection of the field to update using the parameter field.

15

PARAMETERS

PARM 1: User profile field that is to be updated

PARM 2: Buffer name containing data that is to be updated

EDGES

20 0 = Update complete

1 = Update error

SendData

25 This action is used to send data and/or commands to a Host application through the General Purpose Server.

PARAMETERS

PARM 1: Host application function id

PARM 2: Host application subfunction id

30 **PARM 3:** The number of variables to send

PARM 4 - N: The list of variable ids to send.

EDGES

0 = Send completed successfully

1 = Send error (ASI pb)

35 HUP = The caller has hung up.

ReceiveData

40 This action is used to parse the data received back from a Host application through General Purpose Server.

PARAMETERS

PARM 1: Host application function id

PARM 2: Host application subfunction id

PARM 3: The time out in seconds

45 **PARM 4:** The number of variables to receive

PARM 5 - N: The list of variable ids to receive.

EDGES

0 = Receive completed successfully

1 = Host problem

50 2 = No data returned

T1 = Time out (no response)

HUP = The caller has hung up

GetFindData

55

This action is used to look up tables on Host systems. The user is prompted for an entry with PlayPrompt, and then this action is called to accept the caller's entry. When a specified number of keys is entered, a request is sent to the Host for a list of table entries that begins with this input. After the list is returned, a search is made

in the list for a unique match. More keys can be entered by the caller and the search is repeated until a unique match is found within the list. As soon as a unique match is found, the complete entry is placed in the buffer.

For example, for a list of city names that contains Santa Cruz and Santa Clara, the caller enters the first seven keys of these two city names in order to get a unique match. Once the unique match is found, the entire entry for that city name (in the case of Santa Cruz, the entire entry is nine keys) would be placed in the buffer.

PARAMETERS

PARM 1: Host application function id

PARM 2: Host application subfunction id to use to retrieve the data list

PARM 3: Variable id to accept the data into, as in GedData

PARM 4: The minimum length of the input to accept before sending the data request to the Host

PARM 5: Host time out in seconds to wait for a response to the data request.

EDGES

0 = Input successful, and match found

1 = No match

2 = Input incomplete, time out waiting for DTMF input

T1 = Time out. Nothing entered

T2 = Last repeat time

HUP = The caller has hung up.

CallStateTable

This action is used to invoke another state table from within a state table. The other state table is executed until either a CloseSession action or an ExitStateTable action is invoked. CallStateTable is used to implement a series of actions in several state tables. For example, after creating one state table with a series of actions, CallStateTable can be used to execute these actions from other application state tables.

CallStateTable can also be used to implement a menu for a caller to select which application to run. Then each application can be written in its own state table and called from the menu state table.

PARAMETERS

PARM 1: Variable id containing the state table id to execute

PARM 2: Variable id containing the state table release

PARM 3: Variable id containing the state table entry edge.

EDGES

0 to 12 = Edge returned by action ExitStateTable

13 = State table not found

HUP = The caller has hung up.

ExitStateTable

This action is used to return from a nested state table back to the one that called it. The parameter for this action is the variable id that contains an edge value which CallStateTable is to use as its return edge. ExitStateTable ends the execution of the current level nested state table and return to the state table it was called from. CloseSession ends execution of all levels of nested state tables, close the session, and return the state machine back to Idle state. If this action is called from a state table that is not nested, the CloseSession action is executed instead.

PARAMETERS

PARM 1: Variable id containing the return

EDGES None.

Disconnect

This action is used to disconnect a caller. Disconnect is used only for specific purposes because a normal end-of-session occurs when the caller hangs up. It can be followed by a CloseSession action or other actions.

PARAMETERS NONE REQUIRED.

EDGES

0 = Complete.

CloseSession

This action clears all buffers used in the preceding session. It is used as the last action of a session.

PARAMETERS NONE REQUIRED.

5 **EDGES**

0 = Complete

1 = Not possible. This process is deactivated.

APPLICATION SCENARIOS

10

This section introduces the application scenarios. The first part discusses the functional characteristics of the application scenarios. The second part illustrates the use of these applications.

FUNCTIONAL CHARACTERISTICS

15

The VPACK sends the ABCD signalling bits to the voice card driver where, based on the country-specific signalling translation table, the driver translates the bits into the appropriate state and passes the state to the Channel Process or Node Manager. The out-of-band call information signalling includes, at a minimum, the called number. When this call information arrives or a time out condition occurs, the channel goes off-hook the session begins.

20

If signalling information is unavailable, for example, a not capable, or fails to arrive within the defined time limit after ringing begins, the call is answered and default conditions are assumed. An example might be a customer who wishes to order supplies. The customer is directed to a particular line group, and then navigates through a preliminary voice menu to select the specific VIS application to run. When the caller is connected to his/her application, the application script may present a greeting and prompts for a password. The password is verified by querying the application data base server, and the script proceeds with voice menus, prompting for DTMF responses.

25

Host interaction is carried out by passing all coded DTMF responses, other than navigational responses, to the data base server. The customer-written data base server may field the request directly or simply pass the request and response to a remote data base server. In this manner, the customer may elect to use existing data base applications and interface protocols.

30

APPLICATION EXAMPLES

35

The following sections contain examples of application scenarios VIS application, telephone service application, and workstation applications.

VIS Application - Bus Schedules and Fares

40

This scenario is an example of a voice interactive data base application. In this scenario, a consumer calls a listed phone number for a bus schedule and the fare service. Based on the called number identification, the bus schedule script is loaded. In the following system script answer, the words in caps are variables generated from the VAG.

45

"Hello, this is Global Bus Lines route and fare system. You may back up a step at any time by pressing the start key on your touch tone phone. "To begin, please turn to page THREE of your phone book for the bus terminal location codes. "Using the touch tone keypad on your telephone, please enter the THREE-digit terminal code for the city where your travel begins.

(Customer enters three-digit terminal code.)

50

"You are travelling from BOSTON.

"Please enter the THREE-digit code for the location where your travel will terminate.

(Customer enters three-digit terminal code.)

55

"You will be travelling to ATLANTA."

"Please enter the day you wish to travel. One is Monday and seven is Sunday.

(Customer enters one-digit day code.)

"You will be travelling on TUESDAY.

"What time of the day do you want to leave BOSTON? Please enter up to four digits using the 24-hour
5 clock, followed by a # sign.

(Customer enters departure time.)

"Departure time from BOSTON is OH EIGHT HUNDRED hours. Arrival time in ATLANTA is TWENTY FIF-
10 TEEN hours. Fare is EIGHTY-FIVE dollars and SIXTY-EIGHT cents. "Press the pound sign for the next departure or the star key to respecify your trip. If you need further help, press zero then the # sign, and you'll be transferred to a live operator."

Telephone Service Application

15

The flowchart in Figure 10 illustrates the progression of a telephone service application. From a selection of customer-installed options, the customer is requesting her telephone service be suspended for a period of time. After being welcomed to the system at function block 700, she is prompted to enter her telephone number 710, and any pertinent information relevant to the task he is requesting. This information is sent to the Host at
20 function block 720 for verification and implementation.

If the customer has entered invalid information, she is prompted to re-enter the data at or call a help number at function block 730. If the customer has entered valid information, she receives a message to suspend or restore telephone service at function block 740.

After confirming her option to suspend, she is prompted to enter two telephone numbers and the dates she
25 wants her service to be suspended 750. One telephone number is the number she is calling from and the other is a number where she can be reached in case of an emergency.

This data is sent to the Host 760 where it is processed and stored. Then, a message restating the customer's requirements along with a request for confirmation is sent back to the customer as shown in function block 770. At this point, she can confirm the suspension requirements or transfer to an operator.

30

Workstation Applications

This section includes examples of the following workstation applications: answering and message taking, message retrieval, message recording, and message delivery.

35

ANSWERING AND MESSAGE TAKING:

In an answering and message taking forwarded to the VAE, the subscriber's greeting is given to the caller, a message is recorded and stored in the subscriber's mailbox, and the call is terminated.

40

1. Depending on the profile, the script causes either a standard or personalized greeting to be played. If personalized, the greeting must be fetched from the data base. The compressed greeting is retrieved in segments, decompressed, and then sent on the outbound circuit, with successive segments fetched in pipelined fashion.

45

2. After the greeting is played, the inbound circuit is opened to receive voice signals while monitoring for on-hook and DTMF tones.

3. As the caller's spoken message arrives, it is compressed into segments and stored in the data base. As each segment is complete, it is assigned a unique identifier.

4. If a key is pressed, the digit is extracted from the stream and processed against the script. This action is either disregarded or followed by a spoken prompt.

50

5. If the caller hangs up, or if a predetermined time limit is reached, the script causes the ASI to disconnect the telephone circuit. The last message segment is stored and the requisite indexes that are stored in the data base are updated to reflect the addition of the message to the subscriber's mailbox.

6. If the class of service for the subscriber dictates that the arrival of this message requires a notification, the ASI sends a control message to an application in the Host requesting that a notification be scheduled.

55

MESSAGE RETRIEVAL:

In a message retrieval application, the subscriber calls a service number to gain access to the mailbox ser-

vices. The most common service used is the retrieval of messages from the mailbox.

1. The incoming call is received at the ASI, together with the called number identifying the service. The appropriate script is selected and activated.

2. The script causes a prompt to be played to the subscriber requesting a keyed identification and password.

3. The DTMF decoding function in the ASI extracts digits and the control function uses them to ensure that correct steps are followed and the pertinent information is collected. Corrective prompts played, when necessary, and the on-hook tone is monitored.

4. When enough information is collected, the presumed profile is accessed from the data base and the password is verified. Upon verification, a prompt is played giving the options of services available. The subscriber then selects the mailbox retrieval.

5. The ASI requests messages from the data base and constructs a prompt that gives the number of messages. This prompt, along with the service options, is played to the subscriber.

6. The subscriber's keyed response causes the messages to be retrieved from the data base. The subscriber can select from two different options: the longer version where he can listen to the entire message, or the shorter version where he can scan the mailbox for the message he wants to listen to. The flow for each option is:

 O Long version

 – Play message header

 – Play message

 – Delete the message

 – Go on to the next message.

 O Short version

 – Play up to four message headers

 – Select the message number of the message he wishes to listen to, or skip forward to the next four message headers.

7. Message information is retrieved in segments, decompressed, and played back. The retrieval is scheduled sufficiently ahead of the playback rate to avoid interruptions in the regenerated voice. During playback, the subscriber has extensive key-invoked capabilities for review and spacing.

8. After playback of each message, a disposal prompt is given. Choices include deletion, retention, and forwarding. Extensive prompting is available for each option.

9. After retrieving messages, through key-controlled navigation, the subscriber may go to another option on the main menu. If this is done, or if a disconnection occurs, the retrieval transaction is over. The message data base is updated, and the relevant indexes and the subscriber related information is changed to reflect the actions taken.

MESSAGE RECORDING:

In a message recording transaction, the subscriber records a message to be placed in the mailboxes of other subscribers. The subscriber gains access to this service by calling the service access number.

1. The called number activates the script. The subscriber logs on and, using the keypad, selects the recording function from the spoken menu.

2. The steps taken to receive, compress, and store the message are the same as those taken for a forwarded call. Review and edit capability may be invoked with the keypad. The operation may be abandoned at any time by either keyed selection or disconnection.

3. When the end of the message is reached, a prompt that asks for the destination is played. This is keyed as either a single target or a list of targets. The designation may be an explicit phone number or an alias that references a predefined directory. A list is always aliased. The ASI builds the requisite message index entries and causes the updates Host server to be received into the mail data base.

4. Again, the transaction is ended with keypad navigation to other menu options or with the subscriber going on-hook.

MESSAGE DELIVERY:

When a subscriber accesses VAE in direct-access mode, he can record a message. Then, he must specify when to send the message and the destination of the message.

The VAE has two message delivery types: immediate and scheduled. For the immediate delivery type, the default option is the message is delivered to the receiver mailbox as soon as it is sent. For the scheduled delivery type, VAE allows the option of specifying the date (time, day, month) for the message to be sent. The VAE allows

two receiver types. The first is a subscriber identified by a user id (telephone number) or name. The second is a distribution list containing a list of subscribers.

The VAE architecture permits message delivery to a non-subscriber. The system can dynamically create dummy user profiles and mailboxes to store information. Then, through outcalling capability, which is implemented in later releases, the VAE can deliver the message to the receiver. After message delivery, the VAE deletes the dummy information. This function may be implemented in future releases.

SESSION MANAGER DESIGN

The Session Manager design includes the following functions:

- O Channel Processes;
- O Internal State Machine including its internal actions;
- O Application Actions; and
- O National Language Support for the Playprompt action.

This section provides the design specifications for these functions. At system startup, the Node Manager will FORK() and EXEC() a single channel process. The first channel process acts as the Session Manager.

The Session Manager performs all the initialization steps that are common to all the channel processes, then these channel processes to the entire ASI system, and after that, it goes back to being a channel process.

The purpose of the Session Manager is to reduce system startup time and provide a simple means of sharing code space for all the Channel Processes. It does this by performing all initialization code that is common to all Channel Processes, and then uses the FORK() function to generate the required number of processes.

Application Actions

This section describes the Session Manager application actions. They are:

- O PlayPrompt
- O GetKey
- O GetData
- O RecordVoice
- O PlayVoice
- O CheckStorage
- O Disconnect
- O CloseSession.

ACTION_PLAY_PROMPT ROUTINE:

This action builds and plays the voice prompts that are defined in the prompt directory. Prompts consist of the following items:

- O timeout, in seconds, for the user to respond to the prompt at the next GETDATA or GETKEY action;
- O number of times a prompt can be repeated before the GETDATA or GETKEY action returns a timeout; and
- O list of segment id's, system variables, and conditional tests. The conditional tests allow control over what segments and variables are to be played for a particular prompt based on conditions at run-time.

INTERNAL STATE TABLE

The State Machine internal state table provides the State Machine with the basic rules to run the IDLE, ANSWERCALL, PLACECALL, and ENDCALL actions. This table is the foundation of the State Machine. All customer defined state tables are invoked by the ANSWERCALL and actions. Figure 11 shows the internal state table.

System Parameters

The Channel Processes system parameters are:

- O Number of channel processes to run
- O Internal state table
- O Process control block
- O Blocking number for voice I/O 4K buffers

- O Timeout number of seconds before time limit is reached
- O Voice message record stop key
- O Number of records to chop off at the end of recording leading edge of the DTMF key.

5 Error Recovery

All errors are logged and/or the Node Manager is notified. Requested UP is unobtainable and request UP for no SMSI information is also unobtainable. For miscellaneous errors, state tables could be defined to handle most error conditions and would provide flexibility without requiring code changes.

10

NATIONAL LANGUAGE SUPPORT FOR THE PLAYPROMPT ACTION

National Language Support (NLS) setup is a set of rules and programs that play a complex variable in a local custom dependent way. This means that the complex variable is played in a number of different ways.

15 Thus, supporting numerous languages and different language syntax.

The format of data input and output are local custom dependent. Such data includes the numbering system, currency, date formats, time formats and telephone number formats. NLS uses a table-driven design. This eliminates the need to write a new program for each new language or language syntax supported by the system.

Using this approach, the code is independent from the language or syntax; only a new table is required.

20 Examples of complex variables are:

- O Numbers
- O Ordered numbers (first, second, and so on)
- O Date
- O Time
- 25 O Currency
- O Telephone numbers.

How the Complex Variable is Played

30 Figure 12 shows how the complex variable is played. Examples of different language syntax are: Number: 20 is expressed as <20> or <2><10> depending on the country. Date: 6/6/90 is expressed as: <6><19><90>; <June><90>; <6><June><1990>; <the><6th><of><June><19><90>; <national symbol voice><year number>; or <year><6><month><6><day>.

Time: 12 hour time or 24 hour time

35 Currency \$5.25

O <5><dollars><25><cents>

O <5><dollars><and><25><cents>

O <5><yen><25><fen>

Telephone number: 9861234

40 O <9><8><6><pause><1><2><3><4>

O <98><61><234>

NLS Rules

45 The language syntax dependencies are based on a set of NLS rules. NLS rules are created by the System Administration Facility (SAF) and loaded by the Node Manager into SYSPARMS. NLS routines use these rules to break down the complex variables into pieces that play the appropriate voice segment.

The NLS rule has a general form in a character array of: <rule meta character><qualifier><optional voice segment(s)>.... The <optional voice segment> can be used anywhere in the specification array except between the meta character and its qualifier. More than one <optional voice segment> can be used in sequence.

50 The NLS rules are stored in SYSPARMS and loaded by the Node Manager; one set is used per language. Figure 13 is a table depicting the NLS parameter numbers corresponding to the rules set forth below.

NLS Rule Characters and Qualifiers

55

The following are the basic elements used in defining the format of a date, time, currency, or telephone number. This is the general form of all NLS specifications:

<rule meta character><qualifier><voice segid>

- / - meta character for words of numbers
- 9 - <billion> for value of 1000,000,000
- 8 - if there is a single word for 100,000,000
- 7 ... 1 , only if there is a word for the value d - day of the month
- 5 1 - play as PP_VARID_NUMBERS (number)
- 2 - play as PP_VARID_ORDINAL (order)
- 3 - play as PP_VARID_DAYSOFMONTH
- m - month of the year
- 1 - play as PP_VARID_NUMBERS (number)
- 10 2 - play as PP_VARID_ORDINAL (order)
- 3 - play as PP_VARID_MONTHNAMES
- y - year
- 1 - year played as single number
- 2 - year played as century and decade
- 15 3 - year played as decade only
- 4 - use the national year
- w - day of the week; similar qualifier as d
- v<ULONG Segment Id> - Play Segment
- h - hour in the day
- 20 1 - 24 hour format as number
- 2 - 12 hour (AM/PM) format as number
- M - minutes
- 1 play as PP_VARID_NUMBERS (number)
- 2 play as PP_VARID_ORDINAL (order)
- 25 s - seconds
- 1 play as PP_VARID_NUMBERS (number)
- 2 play as PP_VARID_ORDINAL (order)
- the following meta characters always have the qualifier '0'
- \$ - currency unit (dollar)
- 30 c - currency fraction (cents)
- D - play telephone number as digits
- g - play telephone number as number by grouping

Variable Mapping Table

- 35 As mentioned before, variables such as numbers, dates, or telephone numbers are considered complex variables. Based on the corresponding playing rules, these variables are broken down into smaller pieces to play as prompts. This process is accomplished using the PlayPrompt action. To identify these primitive pieces, a variable mapping table is used.

- 40 All primitive variables are located by using two keys: var_id and var_num. Var_id defines the primary data type, such as: PP_VARID_NUMBERS and PP_VARID_MONTHNAMES. Var_num is usually the numerated value within the type. For example, January has the var_num value of one.

- To apply this rule to all 256 possible languages, an NLS variable mapping table is defined in the Host data base for each supported language. This table is organized with three columns:

- 45 O Var_id -such as PP_VARID_MONTHNAMES (short)
- O Var_num -such as January, enumerated with short type
- O segid -voice segment id

- This table is loaded by the Channel Process as part of prompt directory loading for the current language used. The following list shows all the supported variable types:

50

55

<u>VARID</u>	<u>ITEM CONTENTS</u>
PP_VARID_NUMBERS	1 to 20, 30, 40, 50, 60, 70, 80, 90
PP_VARID_HTM	hund. thou. mill. bill.
PP_VARID_MONTHNAMES	Jan. Feb. Mar. . .
PP_VARID_DAYSOFWEEK	Sun. Mon. Tue. Wed. Thu. Fri. Sat
PP_VARID_DAYSOFMONTH	1st, 2nd, 3rd, 4th, . . . 31st
PP_VARID_TIMEOFDAY	am, pm, o' clock, hours
PP_VARID_TIMEOFWEEK	yesterday, today, tomorrow
PP_VARID_ALPHABET	A, B, C, D, . . .
PP_VARID_NOISE	noise type
PP_VARID_SYS_MSGS	voice for system error messages
PP_VARID_ORDINAL	ordinal number

NLS Design Assumptions and Limitations

The following lists the current NLS design features:

O Number (example for American English)

- Base 10 (no 24 = 2 dozens, base 12)
- Only valid entry in NUMSPEC allowed (no 10)
- Primitive Numbers are completed in VARMAP table (0, 1, . . . 19, 20, 30, . . . , 90, etc.)

O Date/time inputs in UNIX format

O Currency inputs are floating point number

O Telephone numbers are always played as numbers

O VARMAP Table

- Completeness is expected from SAF
- No redundancy such as 10 in NUMSPEC in American English or <billion> is not used for <1000><million> British

O NLS rule tables must be correct

O No ordinal unit for <millionth> and <billionth>, VARMAP will have <hundredth> and <thousandth>.

NUMBER SYSTEM:

There are two operations to break a number so it can be used as a segment: division and subtraction. VAG creates the number which allows the Channel Process to do these operations. The number formats assume a base 10, with the most significant digit on the left. The range of numbers is from 0 to 4,294,967,295 (size of ULONG) which means that there are at the most 10 significant digits. This internal data structure is not seen by the VAG user. char numspec(60);

The following are the basic elements used in defining the format of a number:

/ - meta character for words of numbers

9 for <billion>

6 for <million>

3 for <thousand>

2 for <hundred>

v - followed by 4-byte voice segment ids

Example of numspec generated for:

(1) American English

/9v<billion>/6v<million>/3v<thousand>/2v<hundred>

(2) British English

/6v<million>/3v<thousand>/2v<hundred>

This numbering scheme is applicable for both cardinal and ordinal number systems. The only difference between the two systems is using var_id for locating the voice segment. There are two sets of the structure,

one for the cardinal number system, and another for the ordinal number system.

DATE FORMAT:

- 5 The order of the day, month and year along with their voice segment ids are given in this structure. Date Specification:
- datefmt: specifies the play order of day, month and year. If national year used, then SYSPARM 'national_year' contains the base year.
 - d - day of month
 - 10 1 play as PP_VARID_NUMBERS (number)
 - 2 play as PP_VARID_ORDINAL (order)
 - 3 play as PP_VARID_DAYSOFMONTH
 - m - month of year
 - 1 play as PP_VARID_NUMBERS (number)
 - 15 2 play as PP_VARID_ORDINAL (order)
 - 3 play as PP_VARID_MONTHNAMES
 - y - Year
 - 1 year played as single number
 - 2 year played as century and decade
 - 20 3 year played as decade only
 - 4 use the national year
 - w - Day of Week; similar qualifier as d
 - v<ULONG Segment Id> - play segment

The following example shows how the date, 6/6/90, is played.

<u>specification</u>	<u>played</u>
m3d2y2	<June><6th><19><90>
m3d2y3	<June><6th><90>
d2m3y1	<6th><June><1990>
<the>d2<of>m3y2	<the><6th><of><June><19><90>
<nsv>y4<year>m1<month>d1<day>	
is played as:	
<national symbol voice><year number><year><6><month><6><day>	

TIME FORMAT:

- The play order of time units and their voice segment ids are given in this structure.
- Time specification
- 45 char timefmt(20);
- specify the order of playing for time elements
- h - hour in the day
 - 1 - 24 hour format as number
 - 2 - 12 hour (AM/PM) format as number
 - 50 M - minutes
 - 1 play as PP_VARID_NUMBERS (number)
 - 2 play as PP_VARID_ORDINAL (order)
 - s - seconds
 - 1 play as PP_VARID_NUMBERS (number)
 - 55 2 play as PP_VARID_ORDINAL (order)
 - v - if voice needed to insert in between, then next four bytes containing voice segment id.
- The following example shows how the time, 11:30:24, is played.

	<u>specification</u>	<u>played</u>
	h1M1<and>s1<second>	<11><30><and><24><second>
5	h2M1	<11><30><am>

CURRENCY SPECIFICATION:

Two monetary units are allowed; they are dollars and cents. The amount is played as a number with the voice segments of monetary units to make up the currency.

```

Currency specification
char moneyfmt(20); /* how to play dollar/cent */
15 short dollar2cent; /* converting ratio */

- moneyfmt:    this field defines the play order of the amount and their
                monetary unit voices.
20 $    play the dollar amount

0    no qualifier, always played as numbers
25 c    play the cent amount.
0    no qualifier, always played as numbers
v    insert a voice in between, followed by its 4 bytes segment ID.
- dollar2cent: conversion ratio from dollar to
30 cent. For American 1 US dollar = 100 cents, and this value is 100.

```

This is a SYSPARM.

The following example illustrates how the amount of currency, \$5.25, is played.

	<u>specification</u>	<u>played</u>
	\$<dollars>c<cents>	<5><dollars><25><cents>
	\$<dollars><and>c<cents>	<5><dollars><and><25><cents>
40	\$<yen>c<fen>	<5><yen><25><fen>

calling sequence: short play,currency(float money) return true if play is successful. The VAG provides all information except the voice segment id for dollar(s) and cent(s) which will be initialized by the Channel Processor.

TELEPHONE SPECIFICATION:

The play grouping method of a telephone number is given in this structure. Within the group, the telephone number is played as numbers char phonespec(20) = DODODO DODODO DODODO calling sequence: short
 50 play_phone_number(char *phone)

55

- use to specify the rule to play a telephone number
 - phonespec: D, g, or v
 - 5 D - play as a digit, followed by a zero or additional D's for additional digits
 - 0 no qualifier
 - g - play as a group, followed by a zero or additional g's
 - 10 0 no qualifier
 - v - insert a voice between play. This must be followed by a 4-byte voice segment id.
 - 15 ' ' - used as a separator.
- Given phone number 4085546888:

<u>specification</u>	<u>played</u>
DODODO DODODO DODODODO	
	<4><0><8><5><5><4><6><8><8><8>
g0g0g0 g0g0g0 g0g0g0g0	<408><554><6888>

STATE TABLE MANAGER DESIGN

The purpose of the State Table Manager is to provide state tables to the Session Manager program in to control the progression of a call. The State Table Manager is designed to receive messages from a request queue, send a DPRB to the Data Base Interface Manager requesting a state table, receive the notification that the state table has arrived, and notify the requestor. The program is designed to handle multiple requests, concurrently, and is configured to handle invalid requests and tables that cannot be retrieved.

The State Table Manager program also responds to requests from the Node Manager to release 4K buffers and requests to invalidate state tables. This section describes the following:

- O Performance considerations
- O Resources
- O The interface for requestors of state tables
- O The pseudo code that defines the State Table Manager Design.

The State Table Manager and the Prompt Directory Manager share a common table structure and control table design. Figure 14 is a block diagram showing the data flow including interfaces with other processes and the control table.

COMMON ROUTINES DESIGN

Because the control table structure is identical (except for the entry control structure), many of the functions required for both the State Table Manager and the Prompt Directory Manager are consolidated into routines that both managers call.

The Control Table consists of a block of memory large enough to contain enough entries to allow for a run-time-determined number of tables in memory at one time. The memory for the table is allocated from the AIX memory pool during system initialization such that the table is contiguous memory. Semaphore operations are used to prevent more than one process from updating the control table at one time.

VAG COMPONENT DESIGN SPECIFICATIONS

This chapter contains the detailed design specifications for the VAG internal components. As shown in Figure 2, these components and their design specifications include:

- O VAG Performance Considerations

- O VAG Resources
- O VAG Server Interface Specifications
- O VAG Use of Motif and X-Windows
- O VAG Global Data Structures
- 5 O VAG Specific Global Variables
- O Front-End Design
- O Prompt Generator Design
- O State Generator Design
- O Voice Generator Design
- 10 O Application Manager Design
- O VAG System Specifications
- O Utilities.

The performance of the VAG components is subordinate to the call processing functions of the system. This requires the VAG components to operate at a lower priority than call processing tasks and to limit its competition for system resources with the rest of the VAE system. The VAG components are expected to operate during normal loading times without degrading system performance.

VAG RESOURCES

20 Total available RAM memory and CPU processing power impact the performance of the VAG components. A large portion of the required memory is allocated for code and data structures for Motif and X-Windows. Both Motif and X-Windows are window interface managers for AIX. Future versions of these products are expected to provide shared libraries to reduce the memory requirements for each task.

Both Motif and X-Windows make extensive use of allocated storage. Because of design decisions to transfer complete files to the VAG components instead of accessing individual records and due to the way that the Motif scrollable lists function, large amounts of memory will be allocated, dynamically, by the application. The amount of memory allocated will depend on the actual size of the data base files that are edited.

The VAG programs are not expected to be locked like call processing tasks, but instead, operate as a standard-demand paged virtual memory process. This reduces the demand on the system's RAM memory that is required to run the VAG components. The VAG components must release any resources shared with call processing as rapidly as possible, for example 4K buffer blocks.

The displaying of graphics is CPU intensive because the X-Windows low graphics functions perform floating calculations. So, the floating point processing performance of the target platform has a marked impact on the display speed.

FRONT-END DESIGN

The purpose of the Front-End design is to enable or disable a security access level based on the user's id and password. At the beginning of a work session, from the VAE top-level default screen, the Front-End design allows the user to select the Password pop-up box only.

When the user enters his user id and password in the Password pop-up box and selects APPLY, the Front-end design allows the user to access only those menu that are assigned to him in the Administrator Profile. The password is not displayed as the user enters it. The enabling or disabling of the security access level is controlled by setting the sensitivity of specific menu buttons. Sensitivity also determines whether keystrokes or mouse actions can invoke the function associated with the button.

PROMPT GENERATOR DESIGN

50 The purpose of the Prompt Generator is to allow a system user to create, update and delete the prompts executed by a channel process. The Prompt Generator provides a graphic user using Motif, to accomplish these tasks. The Prompt Generator creates, updates, and deletes prompts through a DPRB interface with the Data Base Interface Manager.

The Prompt Generator functions at three levels. The first level creates and updates prompt directories. Each prompt directory consists of prompt directory parameters and a list of prompts. The second level creates, updates, and deletes individual prompts in a single selected prompt directory. Each prompt consists of prompt parameters and a prompt body. The third level creates, updates, and deletes the references to the voice segments, prompt variables, and tests that define the prompt body in a single selected prompt.

EDITING VOICE SEGMENTS

To edit a voice segment, a user selects the voice editor text editor panel as shown in function block 800 of Figure 16. The user selects voices in function block 810 and is presented with the list of voice segments which are retrieved by the data base function. In function block 810, the user finds a segment of interest using search or scrolling, and using a mouse, clicks on a segment of interest and selects the "Mod Voice" by clicking on the selection area on the same panel as depicted in function block 830.

The voice application enabler performs the following steps in preparation for editing:

- a. presents the Voice Editor Panel;
- b. requests the allocation of two voice channels on the voice hardware;
- c. places the channels in wrap mode; one for decompression and the other in "clear channel" (i.e., uncompressed) mode;
- d. locates and writes the compressed voice segment to voice channel in decompression mode, and reads from the wrapped channel in clear channel mode. Both the compressed and decompressed voice segments are retained in memory (RAM);
- e. as it is received from the decompression process, the wave form of the decompressed voice segment is displayed in the voice editor panel; and
- f. the duration of the voice segment in playing seconds is calculated from the its physical length and displayed on the editor panel.

A user selects the "Set" action from the editor panel using the mouse as indicated in function block 850 in Figure 16. Also with the mouse, he marks the beginning and end of the digitized voice segment he desires to modify as highlighted in function blocks 860 and 870. The physical locations in the compressed and uncompressed voice segment are calculated and the MARK'd section is highlighted. These positions are always rounded to the nearest twenty ms. boundary, which in the current implementation are thirty-two and one-hundred- sixty bytes, respectively. The user can then, delete the marked area as shown in function block 880 and 890 and save the resultant as depicted in function blocks 900 and 910.

Alternatively, the user may play the marked portion of the segment by selecting "Play" on the action bar of the Editor Panel. In this case the voice is written to the voice hardware channel which is in clear channel mode and, in turn, is transferred to a headset or speaker. Playing the voice information allows the user to verify that the correct portion of the segment has been selected.

The marked portion may be copied to another segment or deleted by selecting the appropriate actions with the mouse. If the marked portion is deleted, the voice editor rewrites the internal buffers of both the compressed and uncompressed versions of the voice segments, and rewrites the digitized wave form to the panel, resizing the time scale to fit the panel.

If a user desires to copy a whole or portion of a segment into another segment, the first segment is marked, and the user switches to the "Other Window" and INSERTS the COPY'd portion into a MARK'd location of the 2nd segment. The segments are then re-written to their respective buffers and the wave-form panel is re-generated from the uncompressed form of the data.

When editing of the voice segment is complete, the user selects "Save" from the Action Bar. This action saves the edited compressed form of the segment by writing it in a file, thereby replacing the original. Thus, editing is accomplished without successive decompression and recompression of the stored voice data and without successive distortion in the voice as a result.

PLAYING VOICE PROMPTS

Preparation of the prompt

To prepare a voice prompt the user starts the Voice Application Generator (VAG) by selecting "VAG" with the Mouse on the Motif Menu Bar as shown in function block 1000 of Figure 17. The user is presented with a pull-down menu with 3 options: States, Prompts and Voices. Then, the user positions the cursor over the "Prompt" and presses the mouse button to signify selection of the function used to define the prompts he wishes to use for his voice application as shown in function block 1010.

The user is presented with the Prompt Directory Editor Panel. The Prompt Directory Editor panel displays a list of all the currently-defined prompt directory entries. By selecting "Options" the user can choose to display or work with alternative versions of the prompt which may have been defined in other languages. Then, the user selects "Add", as shown in function block 1020 from the Prompt Editor Panel. The input subpanel appears on the right of the Editor Panel.

On the input subpanel, the user enters Prompt ID number and Purpose as shown in function block 1030,

then clicks on "Apply". Then, the user is presented with a list of voice segments available, and an expanded prompt list to the right of the list, and a selection of relationship operators as discussed earlier. The user selects a voice segment or variable, as shown in function block 1040, and optionally a conditional test which allows the referenced segment to be conditionally played depending on the value of the data in a defined variable (e.g., today's date or time, value entered by a caller, etc.). Complex IF..THEN..ELSE logic is also supported.

If the segment selected is a variable, a pull-down menu is presented which defines the manner in which the data stored in the variable is to be vocalized (i.e., PLAY AS..). The user selects how the variable (segment) is to be vocalized (e.g., digit, number, ordinals, currency, date, time, etc.) as shown in function block 1050. Variables used as prompt segments are stored as a combination of the pointer to the variable data plus codes defining how the data will be played.

The user can continue the process discussed above, stringing prompt segments together, as shown in function block 1060, until the prompt is complete. Then, the user clicks on SAVE, saving the prompt for use in the application as depicted in function blocks 1065 and 1075.

15 Playing Prompts with Complex Variables

Application Script is started by a telephone call to the system as shown in function block 1080 of Figure 18. The script references prompts defined during the application development process as shown in function block 1090. Upon execution of the "PLAY PROMPT" action in the script, as depicted in function block 1095, the VAE SESSION MANAGER steps through the list of segments defined by the prompt. Each type of segment (i.e., static voice or variable) is played according to its "Play AS" specification by passing control to a function defined for that "Play As" type. A flow diagram of this process is illustrated in Figure 15.

Actual vocalization is performed by reference to system-level segment primitives (e.g., "hundred", "p.m.", etc.) which are derived from the rules established for each language available in the system as depicted in function blocks 1100 to 1115. For example, currency and time are vocalized according to national custom. These primitives are selected, in effect, by parsing the value of the variable according to these rules. Upon completion of a prompt, control passes to the next script action.

Language-Independent Voice Applications

To define a new application in another language, the system administrator selects the NLS Editor from the main menu as depicted in Figure 19, function block 1200. The Editor panel is presented, with a list of currently-defined languages. English is the base (default) language.

The user selects "Options - New Languages", as shown in function block 1210, and enters the name of the new language to be defined, as illustrated in function block 1220, and clicks on "OK". Then, the system copies the english-based language files into a shadow data base. The system then remaps the display keyboard by selecting "Keyboards" on the NLS Editor Panel. A list of available keyboard mappings is displayed in the Keyboard Selection Panel. The user can narrow the list by selecting "Filter" and the 2-digit code specifying the language of the keyboard of interest (e.g., French) as depicted in function block 1230.

The user then edits the language-based file groups affecting the user (display) interface as depicted in function block 1255 to 1300. These file groups include:

- (a) application developer standard interface terms;
- (b) common VAE user interface terms;
- (c) VAG Editors;
- (d) administrator profile;
- (e) mailbox terms; and
- (f) user profile.

Then, the user edits language configuration parameters (NLS Rules) in the system configuration panel, as shown in function blocks 1310 to 1320, including:

- (a) variable mapping table;
- (b) number format;
- (c) telephone number format;
- (d) currency format; and
- (e) date format.

Application Development (VAG)

To develop the application after the language tailoring has been completed, the user selects "VAG" from

the VAE Sign-on Panel, as shown in Figure 20 at function block 1335. Then, the user selects one of three voice application development tools, as shown in function block 1345, from the pull-down menu:

- (1) "State Generator" to define application scripts;
- (2) "Prompt Generator" to define prompts to be invoked by the scripts; or
- (3) "Voice Generator" to record, playback, and edit voice segments used in the prompts.

From the application development tool menu, the user selects "Options" in the selected tool at function block 1355, then "Language" in the pull-down menu at function block 1365. Then, the user selects language from the list of defined languages in the pull-down menu. The selected language will be used during the development session. The user can develop applications in her choosed language, including scripts, prompt definitions, and voice prompt segments as shown in function block 1375.

NLS Execution

A script is prepared using VAG as discussed above and is invoked using appropriate signalling (i.e., DID, SMSI, dedicated channel, etc.) interfaces. The script executes using NLS rules for vocalizing prompts. The rules are table-driven and defined in the System Configuration section as discussed above. There is one set of rules per language. The set used is the one corresponding to the active language at the time of script execution.

As discussed in greater detail above, the NLS general form is:

<rule meta><qualifier><optional voicesegments>...

Rule meta characters and qualifiers include the following:

<u>Meta Char</u>	<u>Qualifier</u>	<u>Word</u>
/	9	"Billion"
	8	100,000,000
d	1	Play as number ("one")
	2	Play as ordinal ("first")
	3	Play as days of month

etc., (there are meta characters for the year, the time, and for currency)

Complex variables, such those above are, thus, broken down into smaller "primitive" pieces to play as individual words or phrases, according to the NLS rules. These primitives are identified according to a variable mapping table, for example:

<u>Variable ID</u>	<u>Variable Value</u>	<u>Segment ID</u>
MONTHNAME	1	Pointer to "January"

An execution logic NLS flow chart is provided in Figure 21. An incoming call is detected as suggested in function block 1400. The detection invokes an appropriate script that generates the voice prompt playing as pointed out in function block 1410. To execute the script, the prompt definition and the NLS rules must be retrieved as illustrated in function blocks 1420 and 1425. This information allows the prompt generator to decode the PLAY-AS specifications and decode the complex voice information into primitives that can be played back to the caller as shown in function block 1435 and 1440.

Claims

1. A method for editing compressed voice information, comprising the steps of:
 - selecting a segment of compressed voice information;
 - decompressing the compressed voice segment and displaying the decompressed voice information on a display;
 - marking a portion of the displayed voice information;
 - calculating the location and extent of a portion of the compressed voice segment corresponding to

said marked portion;
 editing the marked portion of the decompressed voice information; and
 correlating the editing actions from the decompressed voice information to the compressed voice information.

5

2. A method as claimed in claim 1, the step of editing comprising deleting said marked portion from said decompressed voice information and the step of correlating the editing actions comprising deleting the corresponding portion of the compressed information.

10

3. A method as claimed in claim 1 further comprising:
 selecting a second segment of compressed voice information;
 decompressing the second voice segment and displaying the second segment of decompressed voice information on a display;
 marking a location within the displayed second segment; and
 copying the marked portion from the first segment to said location in said second segment.

15

4. A method as claimed in any preceding claim wherein said marked portion may comprise the whole of said voice segment.

20

5. A method as claimed in any preceding claim wherein the step of decompressing said stored compressed voice information comprises:
 requesting allocation of two voice channels on voice hardware;
 placing the channels in wrap mode;
 locating and writing the compressed voice segment to the first allocated voice channel to decompress the segment; and
 reading the decompressed voice segment from the second allocated voice channel.

25

6. A method as claimed in any preceding claim, further comprising storing said edited compressed voice information on to a permanent record medium.

30

7. Apparatus for editing compressed voice information, comprising:
 means for selecting a segment of compressed voice information;
 means for decompressing the compressed voice segment and displaying the decompressed voice information on a display;
 means for marking a portion of the displayed voice information;
 means for calculating the location and extent of a portion of the compressed voice segment corresponding to said marked portion;
 means for editing the marked portion of the decompressed voice information; and
 means for correlating the editing actions from the decompressed voice information to the compressed voice information.

35

40

8. Apparatus as claimed in claim 7, the editing means comprising means for deleting said marked portion from said decompressed voice information and the correlating means comprising means for deleting the corresponding portion of the compressed information.

45

9. Apparatus as claimed in claim 7 further comprising:
 means for selecting a second segment of compressed voice information;
 means for decompressing the second voice segment and displaying the second segment of decompressed voice information on a display;
 means for marking a location within the displayed second segment; and
 means for copying the marked portion from the first segment to said location in said second segment.

50

10. Apparatus as claimed in any of claims 7 to 9, further comprising means for storing said edited compressed voice information on to a permanent record medium.

55

FIG. 1

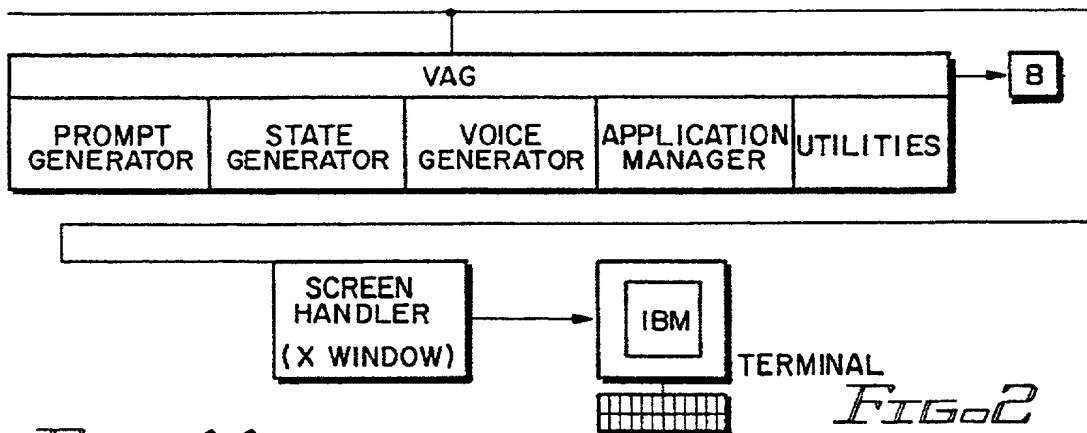
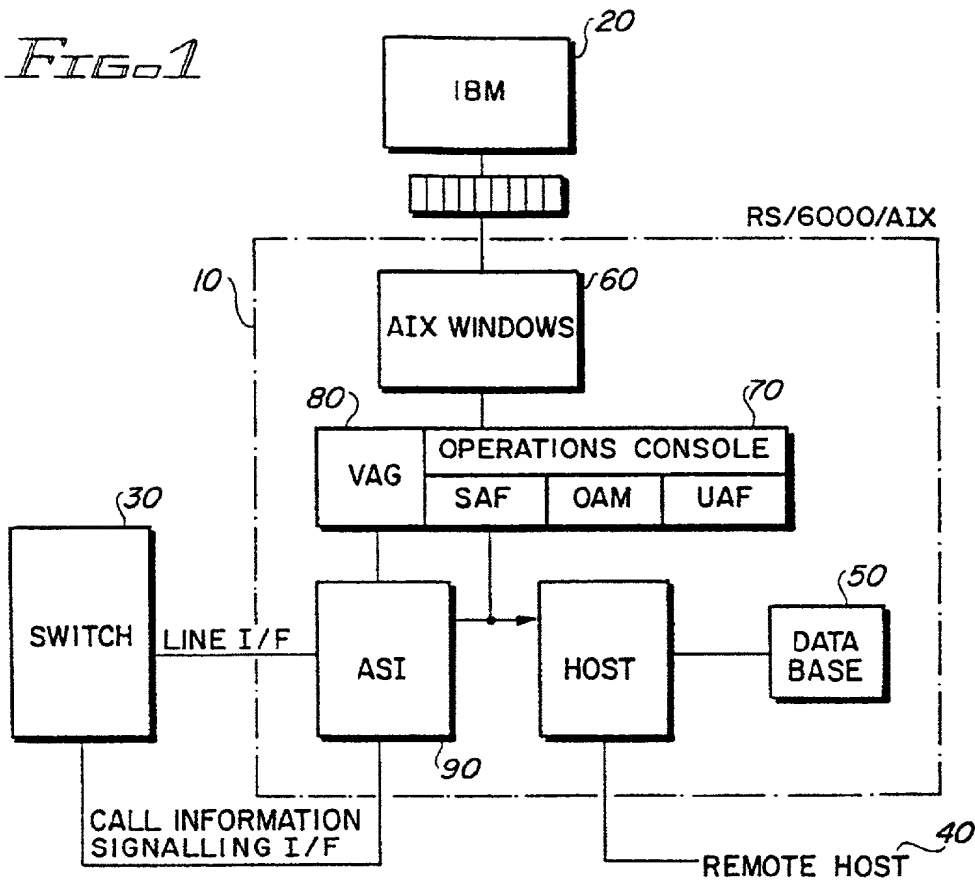
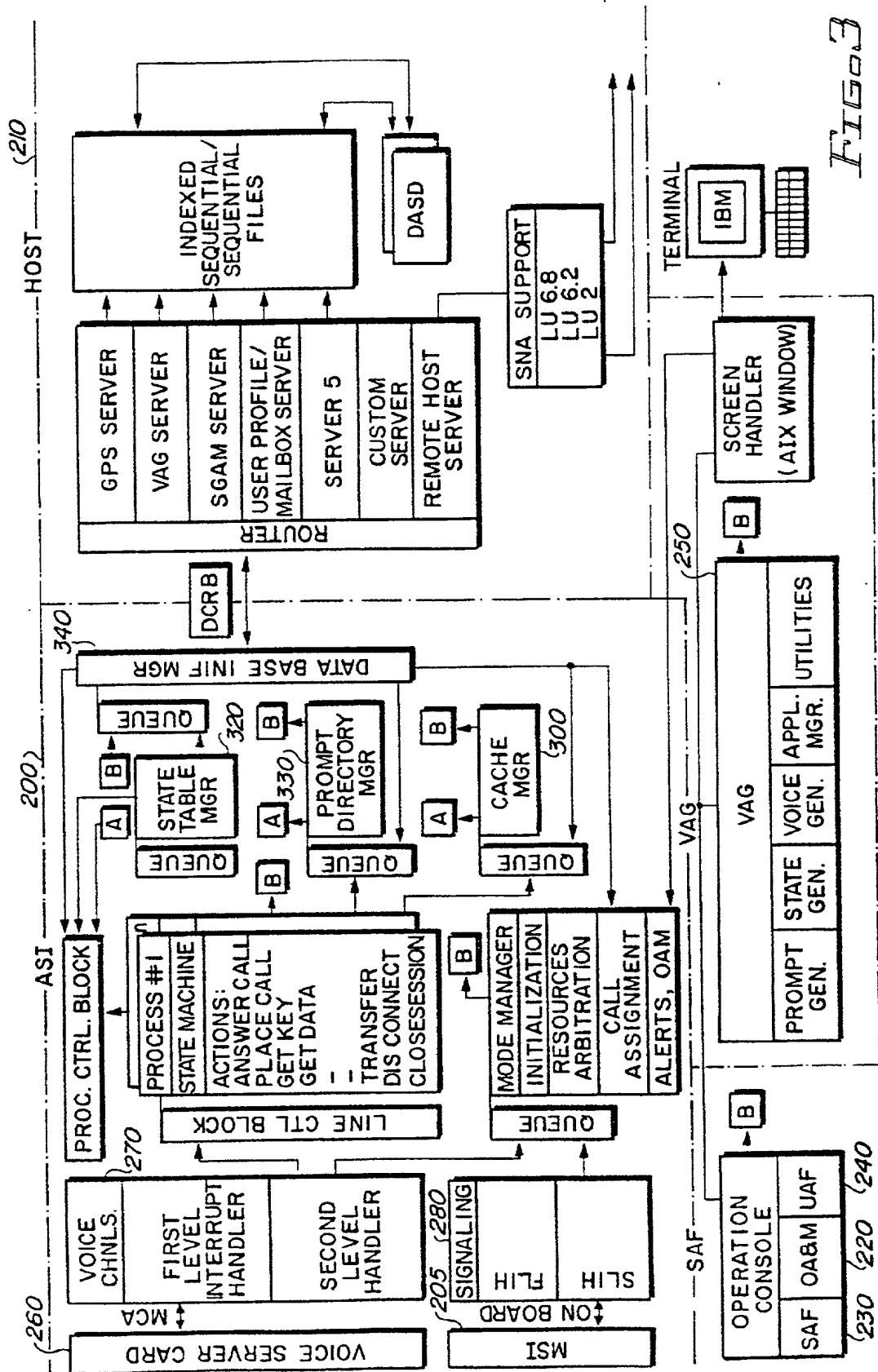


FIG. 11

STATE	ACTION	EDGES															
		0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
0	---	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	
1	IDLE	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	ANSWER CALL	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	4
3	PLACE CALL	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
4	END CALL	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



STATE TABLE NUMBER : 001 RELEASE NUMBER : 00 PURPOSE : BUS SCHEDULE SCRIPT

STATE	ACTION	PARAMETER(S) DESCRIPTION	CONDITION	EDGE	NEXT STATE
1	PP	PLAY GREETING PARAMETER : 1	COMPLETED SUCCESSFULLY VOICE CHANNEL PROBLEM HANG UP DETECTED	0 1 HUP	2 30 40
2	PP	PLAY "ENTER CITY CODE" PARAMETER : 2	COMPLETED SUCCESSFULLY VOICE CHANNEL PROBLEM HANG UP DETECTED	0 1 HUP	3 30 40
3	GD	GET FROM CITY CODE PARAMETERS : 1, 1, 10	INPUT COMPLETE INPUT TOO SHORT INPUT TOO LONG INPUT INCOMPLETE, TIME OUT OCCURRED INPUT NOT STARTED, TIME OUT OCCURRED LAST TIME OUT HANG UP DETECTED	0 1 2 3 T1 T2 HUP	4 50 60 70 -1 30 40
4	SD	SEND GET_CITY CODE TO SERV. PARAMETERS : 100, 100, 1	COMPLETED SUCCESSFULLY, HANG UP DETECTED HANG UP DETECTED	0 HUP	5 30
5	RD	RECEIVE GET_CITY SEGMENT ID PARAMETERS : 1100, 100, 30, 2	COMPLETED SUCCESSFULLY HOST RETURNED INVALID REQUEST HOST PROBLEM TIME OUT WAITING FOR SERVER REPLY HANG UP DETECTED	0 1 2 T1 HUP	6 80 90 90 30
6	PP	PLAY "YOU WILL TRAVEL FROM" PARAMETER : 3	COMPLETED SUCCESSFULLY VOICE CHANNEL PROBLEM HANG UP DETECTED	0 1 HUP	7 30 40

FIG 4

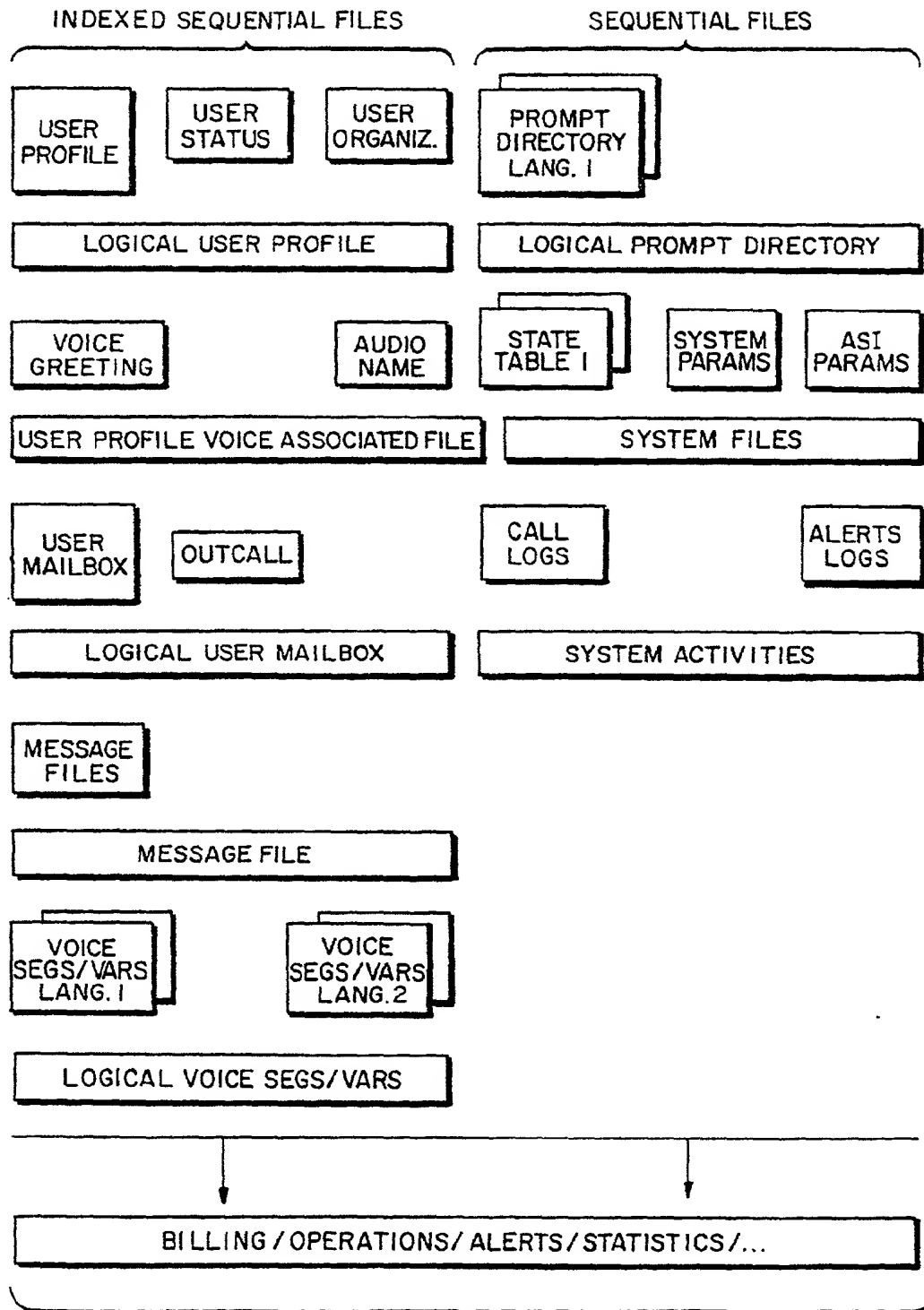


FIG. 5

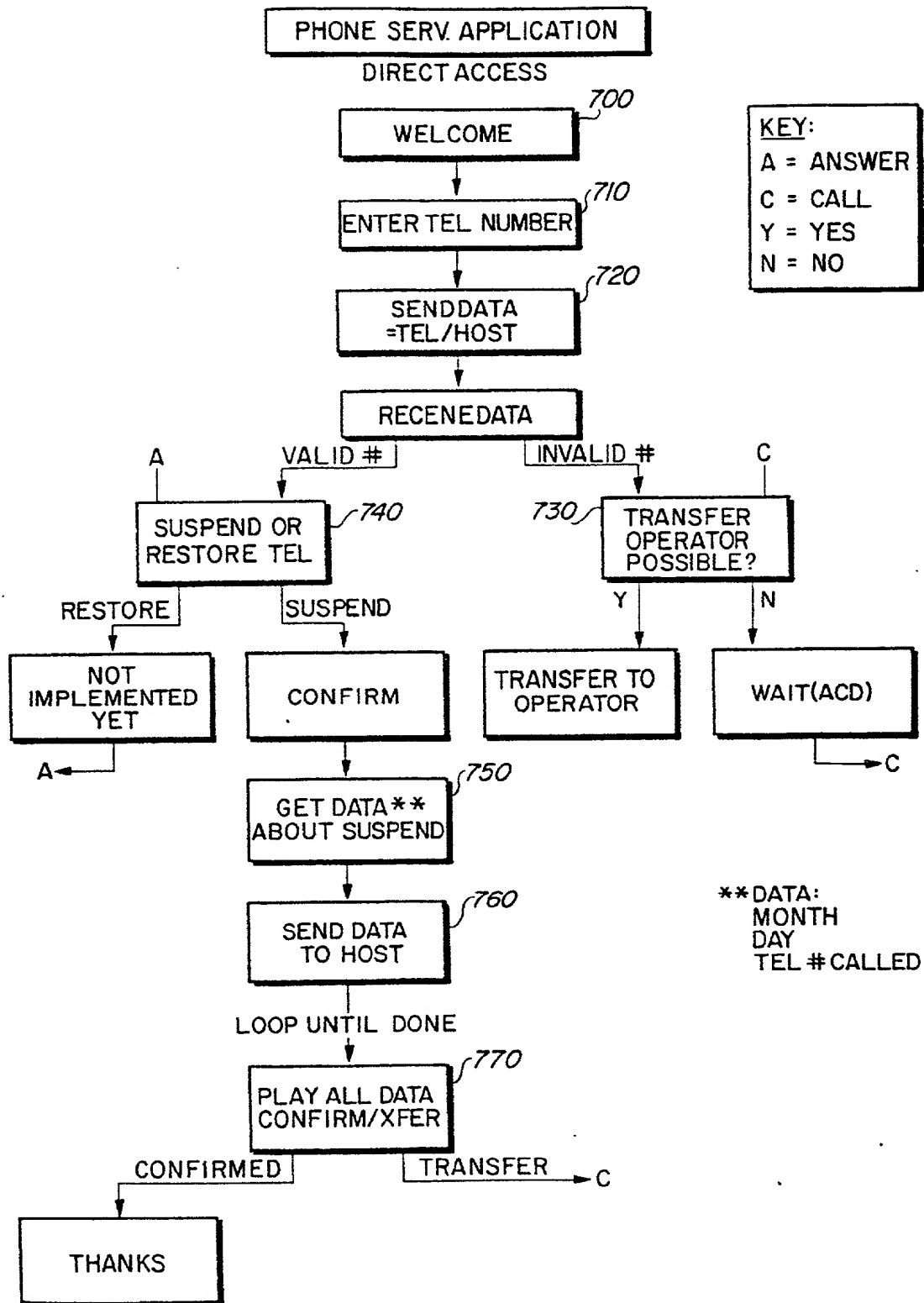


FIG. 10

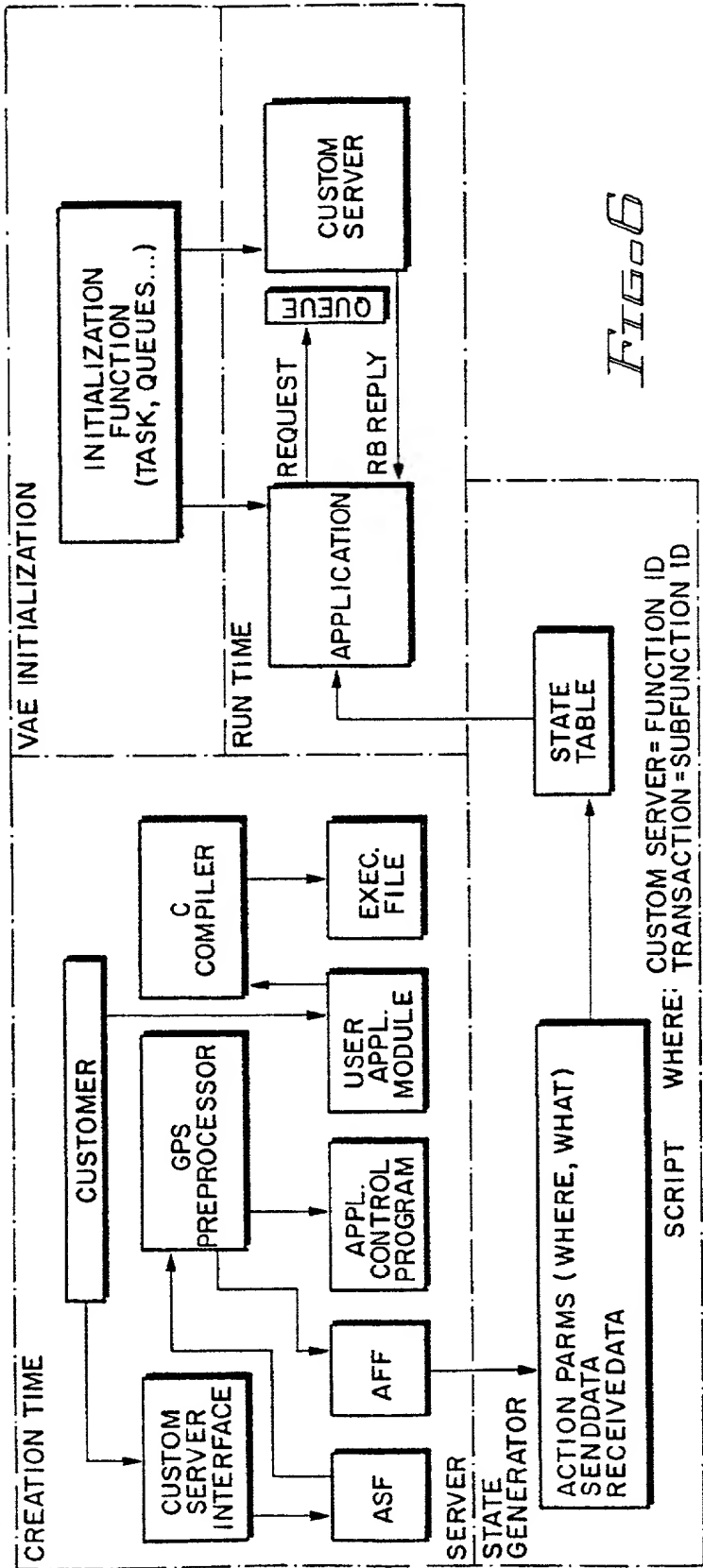


FIG. 6

FIG. 13

NLS PARAMETER NUMBER	DESCRIPTION
1	CARDINAL
2	ORDINAL
3	DATE
4	TIME
5	CURRENCY
6	TELEPHONE
7	BASE
8	DOLLAR-TO-CENT CONVERSION RATIO

VAG Prompt Segment Editor

File Options VAG Help

Add
Delete
Modify
Search

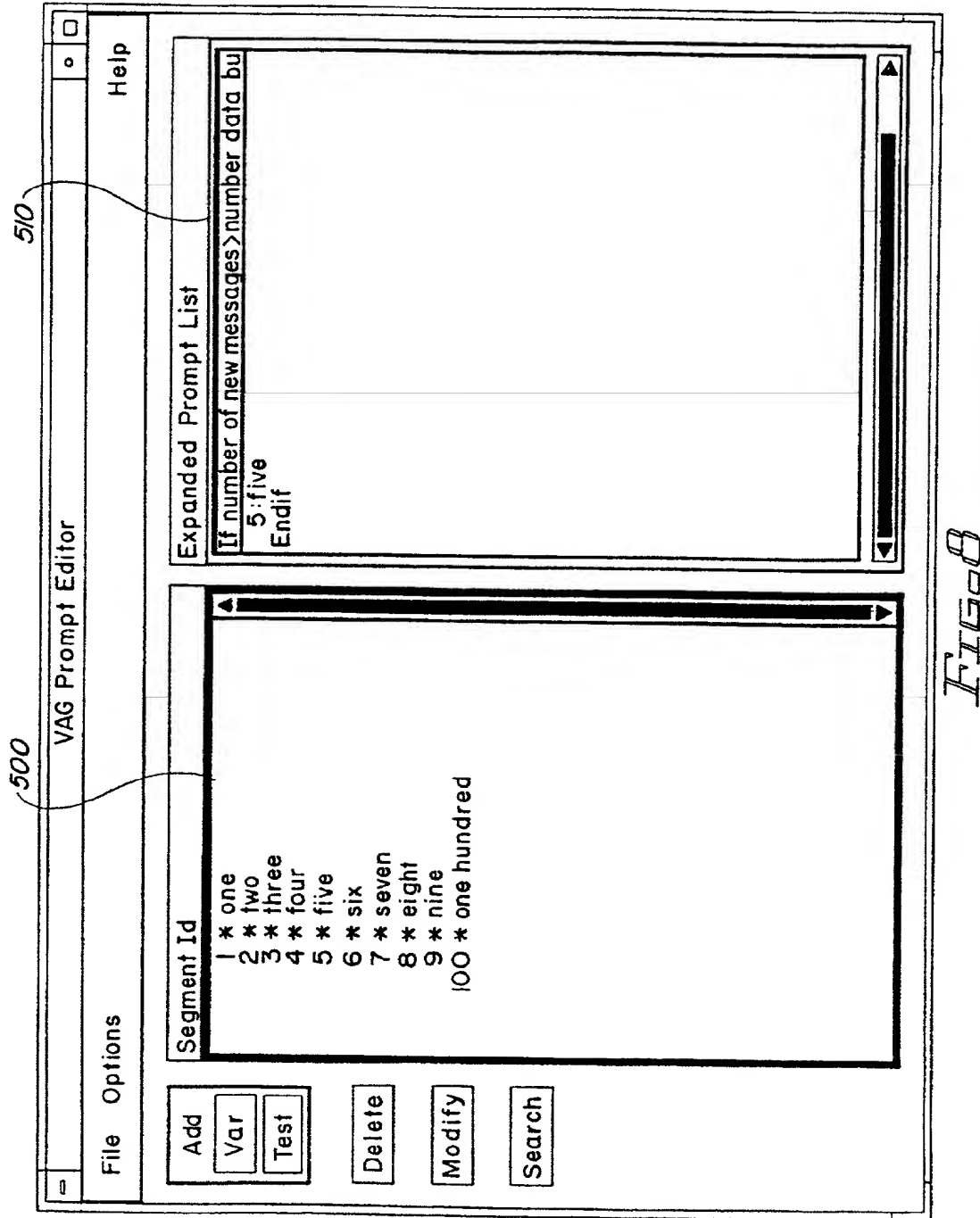
Segment Id	Text
1 * one	
2 * two	
3 * three	
4 * four	
5 * five	
6 * six	
7 * seven	
8 * eight	
9 * nine	
100 * one hundred	

Mod Voice

Selected Prompt Segment
Segment Id Text

Apply
Cancel

Fig 7



600

VAG State Editor

File Options VAG Help

Add

Delete

Modify

Search

00010 Play Greeting

00020 Play User Id Prompt

00030 Retrieve User Id

00040 Play Password Prompt

00050 Retrieve Password

00060 Send User Id/Password to Host

00070 Get Back Verification

00080 See if there's any new mail

00090 Play the next mail message

00100 Continue with next message

00200 Close Session

Selected State

State Number

00010

Purpose

Play Greeting

Action

Play Prompt

Parms

1.2

Edge Values

0	20	1	200	2	
3		4		5	
6		7		8	
9		*		#	
T1		T2		HUP	200

Apply

Cancel

FIG. 9

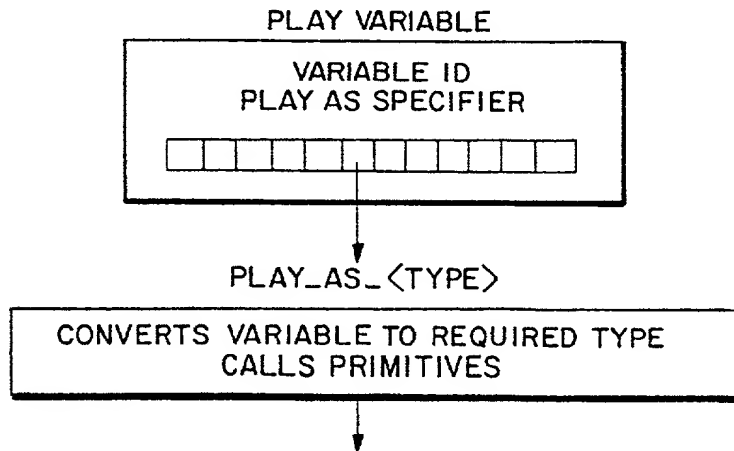
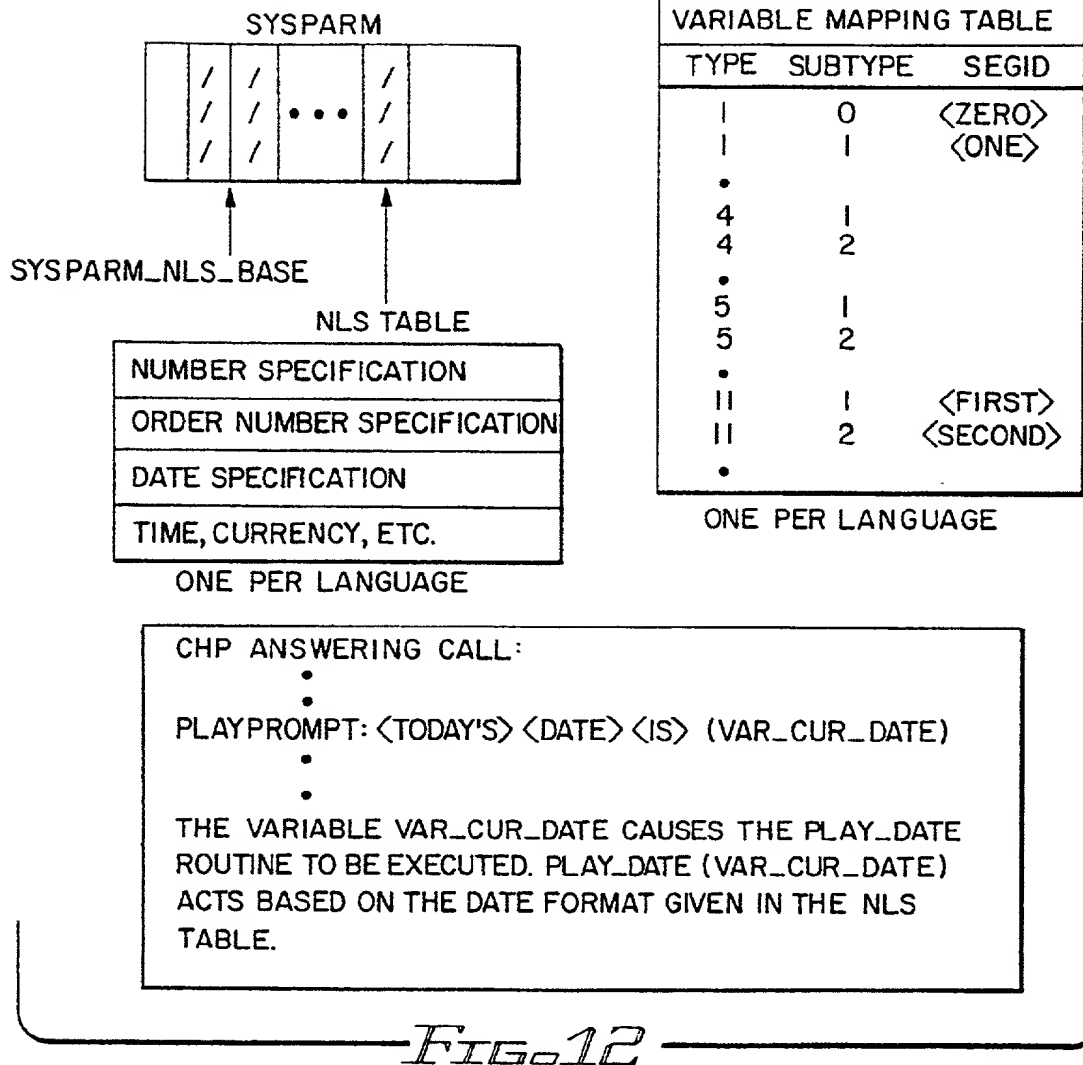


FIG. 15

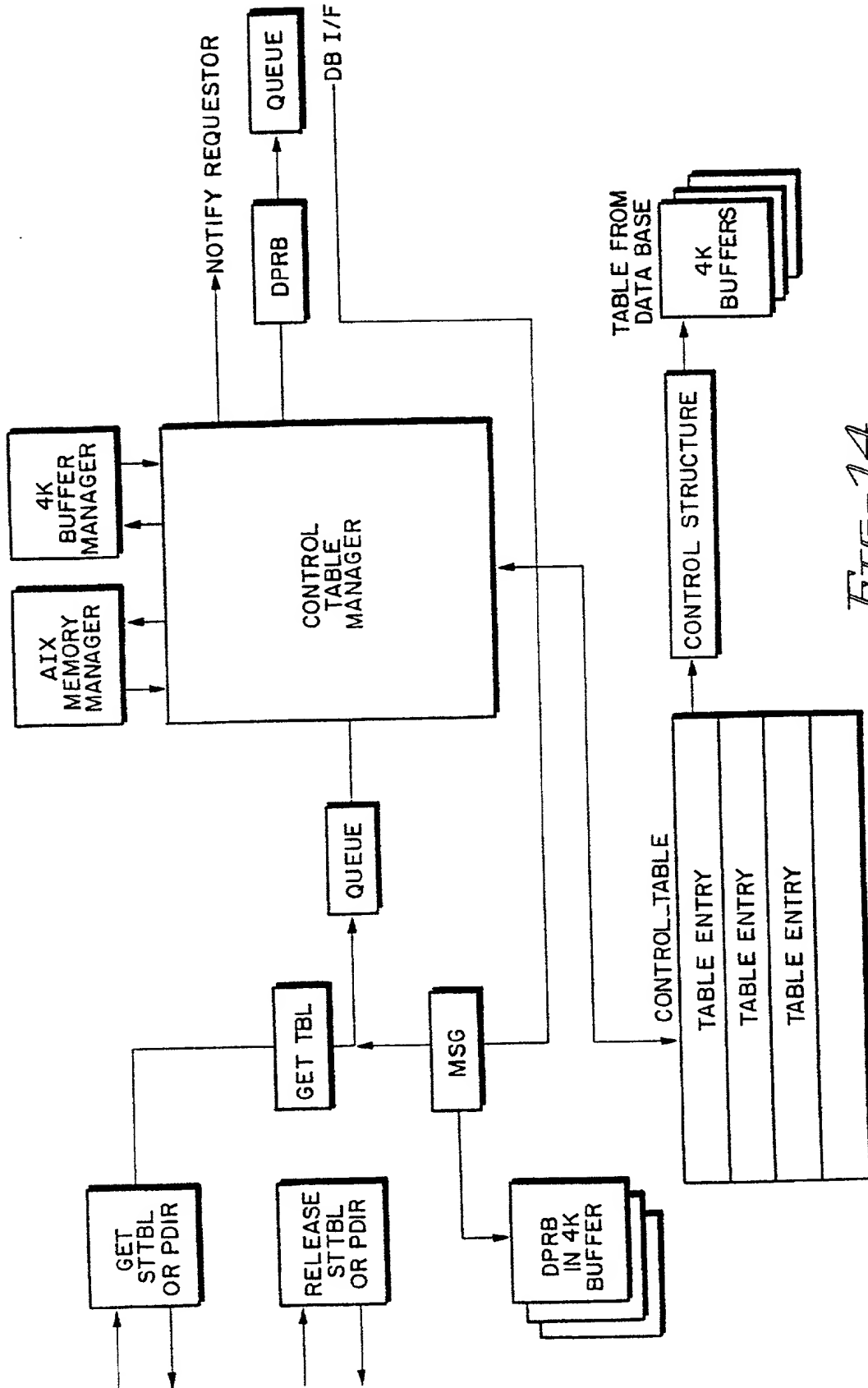


FIG. 14

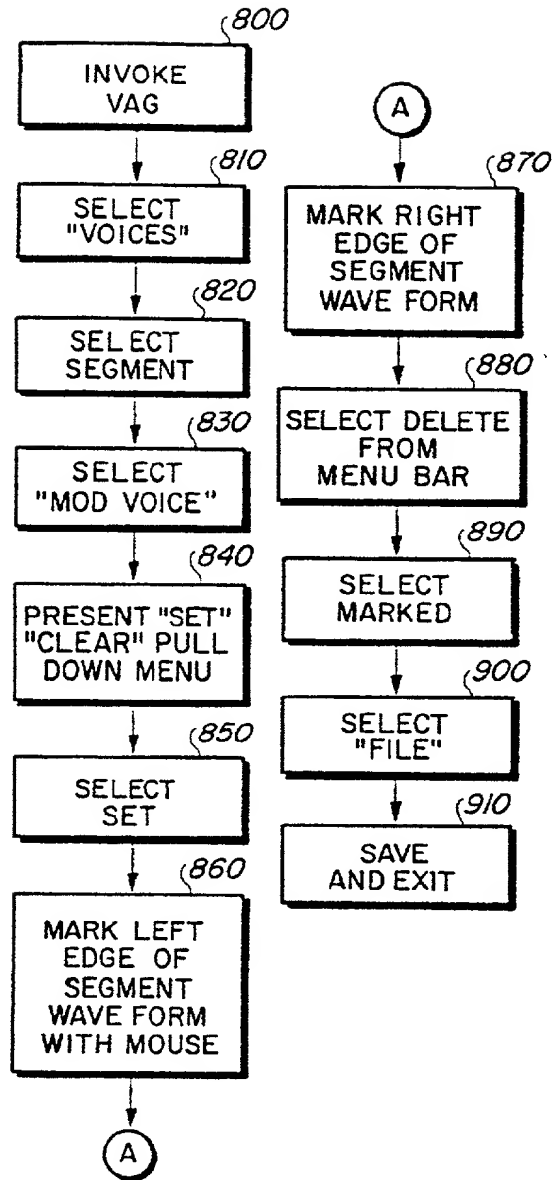


FIG. 16

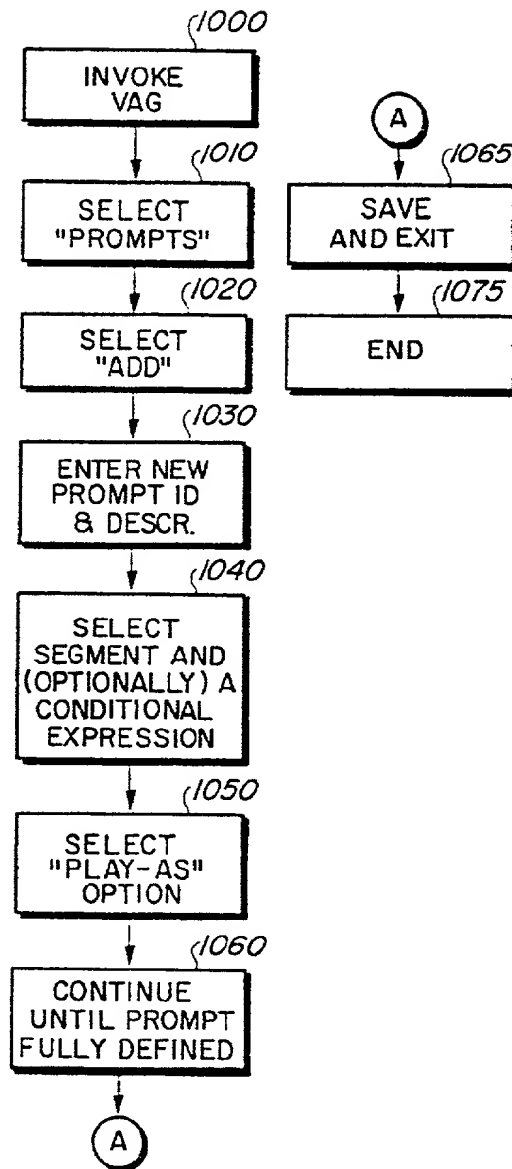
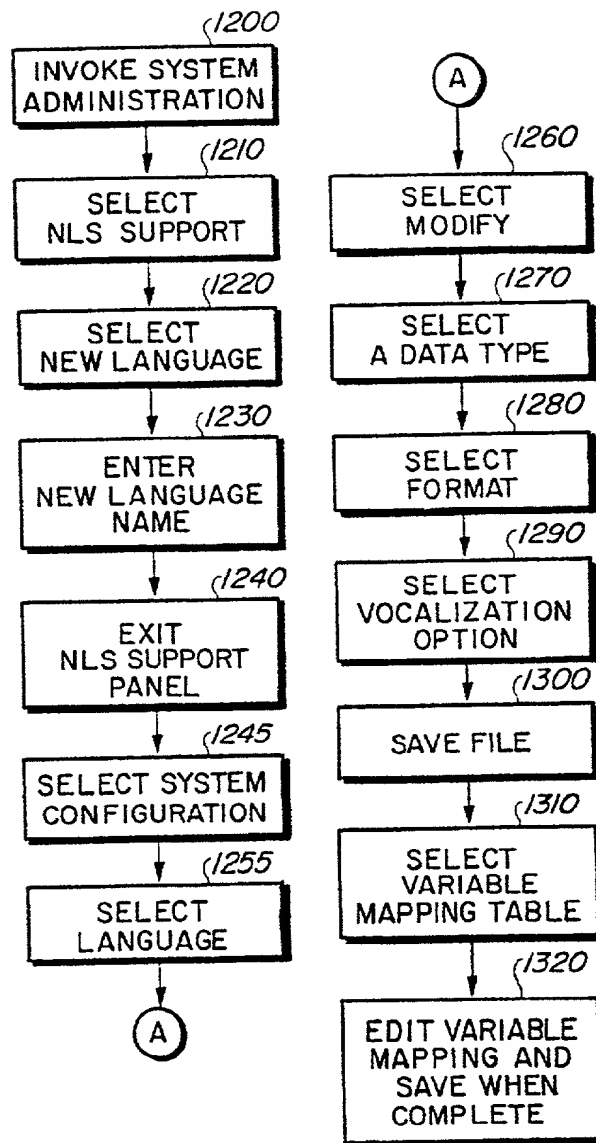
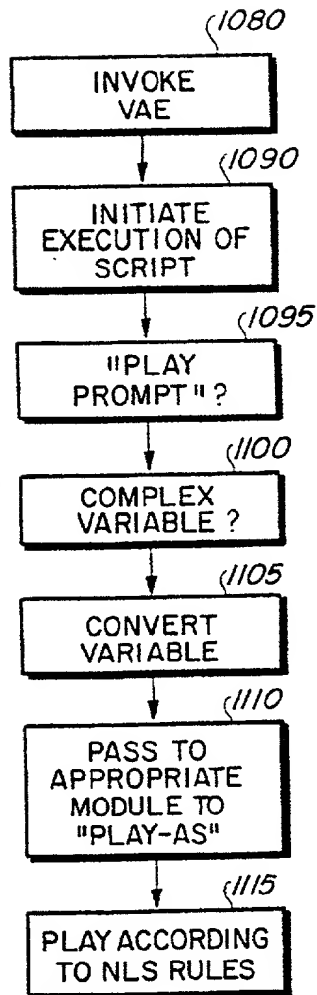


FIG. 17



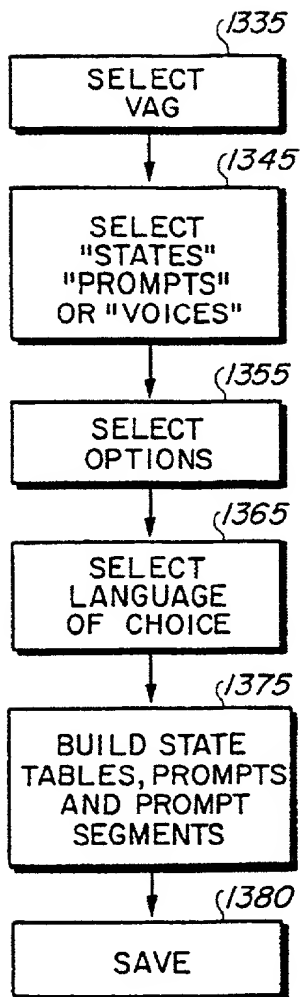


FIG. 20

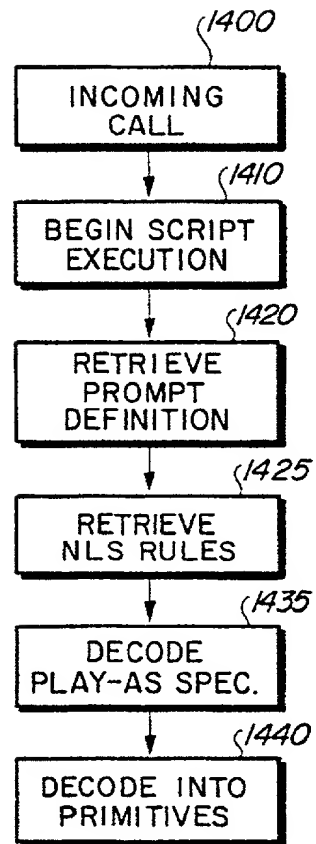


FIG. 21

[illegible]

A method and apparatus are provided for resolving uncertainty in information provided to an information service. A database stores a list of one or more likely responses to an explicit or implicit request for information. Received information from a user in response to a request is compared to one or more of the likely responses in the list to identify such received information. Associated with each likely response is an a priori probability that the response will be provoked by the request. A priori probabilities may be based on, among other things, training with a user or a back-up procedure for resolving uncertainty. In comparing received information to a likely responses, a comparison score is generated. A comparison score is checked to determine whether it is within a range of acceptable comparison scores. If so, received information may be identified. If not, a back-up uncertainty resolution technique may be performed.



(12)

EUROPEAN PATENT APPLICATION

(21) Application number: 92307202.9

(51) Int. Cl.⁵: G10L 5/00

(22) Date of filing: 06.08.92

(30) Priority: 16.08.91 US 746444

(43) Date of publication of application:
24.03.93 Bulletin 93/12

(84) Designated Contracting States:
DE ES FR GB IT

(71) Applicant: **AMERICAN TELEPHONE AND
TELEGRAPH COMPANY**
32 Avenue of the Americas
New York, NY 10013-2412 (US)

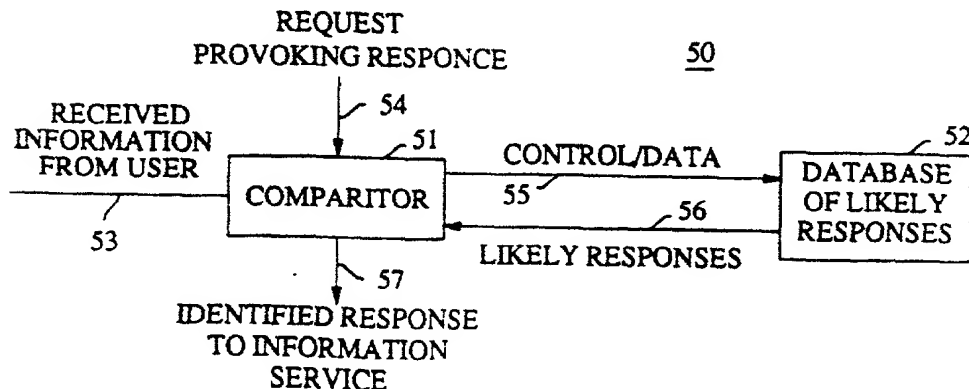
(72) Inventor: **Rabiner, Lawrence Richard**
58 Sherbrook Drive
Berkeley Heights, New Jersey 07922 (US)

(74) Representative: **Watts, Christopher Malcolm
Kelway, Dr. et al**
AT & T (UK) Ltd. 5, Morningson Road
Woodford Green Essex, IG8 0TU (GB)

(54) Interface method and apparatus for information services.

(57) A method and apparatus are provided for resolving uncertainty in information provided to an information service. A database stores a list of one or more likely responses to an explicit or implicit request for information. Received information from a user in response to a request is compared to one or more of the likely responses in the list to identify such received information. Associated with each likely response is an *a priori* probability that the response will be provoked by the request. *A priori* probabilities may be based on, among other things, training with a user or a back-up procedure for resolving uncertainty. In comparing received information to a likely responses, a comparison score is generated. A comparison score is checked to determine whether it is within a range of acceptable comparison scores. If so, received information may be identified. If not, a back-up uncertainty resolution technique may be performed.

FIG. 2



Field of the Invention

The present invention relates generally to information services and in particular to user interfaces for information services.

Background of the Invention

Information services are widely used to provide access to and management of information or data. Examples of information services include financial services, such as those used by individuals to purchase securities or transfer funds; database services, such as those used to store, search for and retrieve information; and telephone services, such as those used to identify and dial telephone numbers. Typically, a user interacts with an information service with the aid of a user interface. The interface may include audio and graphical features supported by an input/output (I/O) device, such as, for example, a personal computer, computer terminal, or telephone.

Information service user interfaces are often described as tree-like in nature, having nodes and branches. The nodes of the tree represent explicit or implicit questions or requests ("requests") for information to be put to a service user. User responses to such requests allow an information service to determine the type of processing and functions desired. For example, a service may request a stock name for which a price quote is sought by a user, or a telephone number which a user desires to dial. The branches of the tree represent paths between successive requests, or paths between a request and a function to be performed by the service.

Information responsive to a request may be provided to an information service by any number of input techniques and associated devices. These include speech through a microphone, a keyboard or key-pad, a pen-like stylus, bar-code or magnetic media scanning, push-buttons, touch-screen technology, etc. Depending on the nature of the information service or the tasks required of the user, one or more of such techniques may be preferred over others. For example, voice entry of information may be preferred in some instances to speed and simplify information service operation for users. Voice entry may also be preferred because there is no alternative I/O device, or because of special needs of a user (e.g., due to a handicap).

As a consequence of the nature or use of an input technique or its associated device, the content of information received by an information service interface in response to a request may be subject to some degree of uncertainty. For example, in the form received from a microphone, the content or meaning of speech signals may not be recognizable by the information service; signals received from a stylus or bar code scanner may be corrupted in some fashion; or, more than one key on a keypad or element in a touch-screen system may be depressed accidentally. In each of these cases, the content of received information is uncertain. Prior to proceeding with service processing, the information service interface needs to address such uncertainties of received information content. In the illustrative case of speech input, the information service interface must perform processing to recognize the content of spoken words such that the information will be in a form useful to the service.

Summary of the Invention

The present invention provides a method and apparatus for resolving uncertainty in the content of information received as input to an information service. Resolution of uncertainty is provided by reference to a database containing likely responses to requests for information. A response is deemed *likely* based on an *a priori* probability that the response will be provoked by a given request. *A priori* probabilities therefore indicate with what information a given user is *likely* to respond when presented with a given request. They may be determined either quantitatively or qualitatively based on, among other things, the nature of the information service or experience with its use.

Information of uncertain content received by the service interface is compared to the likely stored responses for the purpose of resolving the uncertainty. An illustrative embodiment of the present invention may perform the comparison in any of several ways. For example, the received information may be identified as the stored response to which it most closely compares based on a similarity metric. The received information may be *tentatively* identified as discussed above and an information service user be provided with a "right of refusal" of the identified information, to be exercised in the event that the *a priori* probable responses stored in the database do not provide for a reasonable resolution of the uncertainty.

Furthermore, the received information may be identified, tentatively or otherwise, as the first stored encountered response in the database (or portion thereof with which a comparison to the received information yields an acceptable measure of similarity. This technique may be used in conjunction with an ordering of likely responses in the database based on likelihood of use.

- determining the security level of data. The security constraints are used as deviation rules. The architecture for a query processor is shown in FIG. 2. This architecture can be regarded as a loose coupling between a 5 multilevel relational database management system and a deductive manager. The deductive manager is referred to as the query processor. It operates on-line.

An update processor prototype is disclosed. The processor utilizes simple and content-dependent security constraints as guidance in determining the security level of the data being updated. The use of security constraints can thereby protect against users incorrectly labelling data as a result of logging in at the wrong level, against data being incorrectly labelled when it is imported from systems of different modes of operation and against database inconsistencies as a consequence of the security label of data in the database being affected by data being entered into the database. The architecture for the update processor is shown in FIG. 3. This architecture can be regarded as a loose coupling between a multilevel relational database management system and a deductive manager. The deductive manager is referred to as the update processor. It can be used on-line where the security levels of the data are determined during database inserts and updates, or it could be used off-line as a tool that ensures that data entered via bulk data loads and bulk data updates is accurately labelled.

The security level of an update request is determined by the update processor as follows. The simple and content-dependent security constraints associated with the relation being updated and with a security label greater than the user log in security level are retrieved and examined for applicability. If multiple constraints apply, the security level is determined by the constraint that specifies the highest classification level. If no constraints apply, the update level is the Logan security level of the user. The update processor does not determine the security level of the data solely from the security constraints, but utilizes the constraints as guidance in determining the level of the input data.

In the disclosed apparatus and method, all constraints except for the release and aggregate constraints can theoretically be handled during the database update operation. When constraints are processed during the update operation, the update processor will compute the security levels of the data being updated and ensure that the data is stored at the appropriate level.

An MLS DBMS provides assurance that all objects in a database have a security level associated with them and that users are allowed to access only the data which they are cleared. Additionally, it provides a mechanism for entering multilevel data but relies on the user to Logan at the level at which the data is to be entered. The Update Processor will provide a mechanism that can operate as a standalone tool with a MLS DBMS to provide assurance that data is accurately labelled as it is entered into the database. This could significantly enhance and simplify the ability of an MLS DBMS to assure that data entered via bulk data loads and bulk data updates is accurately labelled.

Another significant use for an update processor is in operation with an Inference Controller which functions during query processing. The Inference Controller protects against certain security violations via inference that can occur when users issue multiple requests and consequently infer unauthorized knowledge. The Inference Controller Prototype also utilizes security con-

straints as its mechanism for determining the security level of data. The security constraints are used as derivation rules as they are applied to the data during query processing. Addressing all of the security constraint types mentioned above could add a significant burden to the query processor particularly if the number of constraints is high. To enhance the performance of the query processor, the Update Processor can be utilized to address certain constraint types as data is entered into the database, in particular, simple and content-based constraints, alleviating the need for the query processor to handle these constraint types. We assume that the security constraints remain relatively static, as reliance on the Update Processor to ensure that data in the database remains consistent would be difficult, particularly in a volatile environment where the constraints change dynamically. An additional concern is that database updates could leave the database in an inconsistent state. The Update Processor, however, is designed to reject updates that cause a rippling effect and thus leave the database in an inconsistent state.

A method and apparatus for handling constraints during database design is disclosed. The database design tool is shown in FIG. 4. The constraint generator takes the specification of the multilevel application and outputs the initial schema and the constraints that must be enforced. The database design tool takes this output as its input and designs the database. The constraints and schema produced by the database design tool are used by the update processor and the query processor.

BRIEF DESCRIPTION OF THE DRAWING

FIG. 1 is a block diagram of the integrated architecture of the invention.

FIG. 2 is a block diagram illustrating the query processor of the invention.

FIG. 3 is a block diagram illustrating the update processor of the invention.

FIG. 4 is a block diagram illustrating a multi-level database design tool.

FIG. 5 is a block diagram illustrating constraint generation and enforcement.

FIG. 6 is a block diagram illustrating high-level architecture.

FIG. 7 is a schematic illustration of a constraint structure.

FIG. 8 is a schematic illustration of the major modules of the invention.

FIG. 9 is a block diagram illustrating another high-level architecture.

FIG. 10 is a block diagram illustrating the implementation architecture according to the invention.

PREFERRED EMBODIMENT

The theoretical approach to security constraint processing is first presented. Then we give the implementation details for the query processor, the update processor and the database design tool.

1. SECURITY CONSTRAINTS

1.1 OVERVIEW

Security constraints are rules which assign security levels to the data. They can be used either as integrity rules, derivation rules or as schema rules (such as data dependencies). If they are used as integrity rules, then they must be satisfied by the data in the multilevel database. If they are used as derivation rules, they are applied to the data during query processing. If they are

used as data dependencies, they must be satisfied by the schema of the multilevel database.

We have defined various types of security constraints. They include the following:

- (i) Constraints that classify a database, relation or an attribute. These constraints are called simple constraints.
- (ii) Constraints that classify any part of the database depending on the value of some data. These constraints are called context-based constraints.
- (iii) Constraints that classify any part of the database depending on the occurrence of some real-world event. These constraints are called event-based constraints.
- (iv) Constraints that classify associations between data (such as tuples, attributes, elements, etc.). These constraints are called association-based constraints.
- (v) Constraints that classify any part of the database depending on the information that has been previously released. These constraints are called release-based constraints. We have identified two types of release-based constraints. One is the general release constraint which classifies an entire attribute depending on whether any value of another attribute has been released. The other is the individual release constraint which classifies a value of another attribute depending on whether a value of another attribute has been released.
- (vi) Constraints that classify collections of data. These constraints are called aggregate constraints.
- (vii) Constraints which specify implications. These are called logical constraints.
- (viii) Constraints which have conditions attached to them. These are called constraint with conditions.
- (ix) Constraints that classify any part of the database depending on the security level of some data. These constraints are called level-based constraints.

- (x) Constraints which assign fuzzy values to their classifications. These are called fuzzy constraints.

We will give examples of constraints to each category. In our examples, we assume that the database consists of two relations SHIPS and ASSIGNMENT where SHIPS has attributes S#, SNAME, CAPTAIN, and A# (with A# as the key), and ASSIGNMENT has attributes A#, MISSION, and DESTINATION (with A# as the key). Note that A# in SHIPS and A# in ASSIGNMENT takes values from the same domain. The constraints may be expressed as some form of logical rules. We have chosen horn clauses to represent the constraints. This way we could eventually take advantage of numerous techniques that have been developed for logic programs.

Simple constraints: $R(A_1, A_2, \dots, A_n) \rightarrow \text{Level}(A_{i1}, A_{i2}, \dots, A_{it}) = \text{Secret}$ [Each attribute $A_{i1}, A_{i2}, \dots, A_{it}$ of relation R is Secret] Example: SHIPS (S#, SNAME, CAPTAIN, A#) \rightarrow Level (CAPTAIN) = Secret

Content-based constraints: $R(A_1, A_2, \dots, A_n) \text{ AND } \text{COND}(\text{Value}(B_1, B_2, \dots, B_m)) \rightarrow \text{Level}(A_{i1}, A_{i2}, \dots, A_{it}) = \text{Secret}$ [Each attribute $A_{i1}, A_{i2}, \dots, A_{it}$ of relation R is Secret if some specific condition is enforced on the value of some data specified by B_1, B_2, \dots, B_m] Example: SHIPS (S#, SNAME, WEIGHT, A#) AND (Value (SNAME) = CHAMPION) \rightarrow Level (CAPTAIN) = Secret.

Association-based constraints (also called context or together constraints): $R(A_1, A_2, \dots, A_n) \rightarrow \text{Level}(\text{Together}(A_{i1}, A_{i2}, \dots, A_{it})) = \text{Secret}$ [The attributes $A_{i1}, A_{i2}, \dots, A_{it}$ of relation R taken together are Secret] Example: SHIPS (S#, NAME, CAPTAIN, A#) $\rightarrow \text{Level}(\text{Together}(\text{SNAME}, \text{CAPTAIN})) = \text{Secret}$.

Event-based constraints: $R(A_1, A_2, \dots, A_n) \text{ AND Event}(E) \rightarrow \text{Level}(A_{i1}, A_{i2}, \dots, A_{it}) = \text{Secret}$ [Each attribute $A_{i1}, A_{i2}, \dots, A_{it}$ of relation R is Secret if event E has occurred] Example: SHIPS (S#, SNAME, CAPTAIN, A#) AND Event (Change of President) $\rightarrow \text{Level}(\text{CAPTAIN}, A\#) = \text{Secret}$.

General release-based constraints: $R(A_1, A_2, \dots, A_n) \text{ AND Release}(A_i, \text{Unclassified}) \text{ CONG} \rightarrow \text{Level}(A_j) = \text{Secret}$ [The attribute A_j of relation R is Secret if the attribute i has been released at the Unclassified level] Example: SHIPS(S#, SNAME, CAPTAIN, A#) AND Release(SNAME, Unclassified) $\rightarrow \text{Level}(\text{CAPTAIN}) = \text{Secret}$.

Individual release-based constraints: $R(A_1, A_2, \dots, A_n) \text{ AND Individual-Release}(A_i, \text{Unclassified}) \rightarrow \text{Level}(A_j) = \text{Secret}$ The individual release-based constraints classify elements of an attribute at a particular level after the corresponding elements of another attribute have been released. They are more difficult to implement than the general release-based constraints. In our implementation, the individual release-based constraints are handled after the response is assembled while all of the other constraints are handled before the response is generated.

Aggregate constraints: Aggregate constraints classify collections of tuples taken together at a level higher than the individual levels of the tuples in the collection. There could be some semantic association between the tuples. We specify these tuples in the following form: $R(A_1, A_2, \dots, A_n) \text{ AND Set}(S, R) \text{ AND Satisfy}(S, P) \rightarrow \text{Level}(S) = \text{Secret}$ This means that if R is a relation and S is a set containing tuples of R and S satisfied some property P , then S is classified at the Secret level. Note that P could be any property such as "number of elements is greater than 10."

Logical constraints: Logical constraints are rules which are used to derive new data from the data in the database. The derived data could be classified using one of the other constraints. Logical constraints are of the form: $A_i = \Rightarrow A_j$ if condition C holds. This constraint can be instantiated as follows: The location of a ship implies its mission if the location is the Persian Gulf.

Other constraints:

There are several other types of constraints which could be incorporated into our design fairly easily. These include level-based constraints and fuzzy constraints. We describe them below.

Level-based constraints: $R(A_1, A_2, \dots, A_n) \text{ AND Level}(A_i) = \text{Unclassified} \rightarrow \text{Level}(A_j) = \text{Secret}$ (The attribute A_j of relation R is Secret if the attribute A_i is unclassified) Example: SHIPS(S#, SNAME, CAPTAIN, A#) AND Level(SNAME) = Unclassified $\rightarrow \text{Level}(\text{CAPTAIN}) = \text{Secret}$

Fuzzy Constraints: Fuzzy constraints are constraints which use fuzzy values. They can be associated with any of the other types of con-

straints. An example of a fuzzy constraint which is associated with a content-based constraint is given below. $R(A_1, A_2, \dots, A_n) \text{ AND COND-Value}(B_1, B_2, \dots, B_m) \rightarrow \text{Level}(A_{i1}, A_{i2}, \dots, A_{it}) = \text{Secret}$ and Fuzzyvalue = r (Each attribute $A_{i1}, A_{i2}, \dots, A_{it}$ of relation R is Secret with a fuzzy value of r if some specific condition is enforced on the values of some data specified by B_1, B_2, \dots, B_m)

Example: SHIPS(S#, SNAME, CAPTAIN, A#) AND (Value(SNAME) = CHAMP 10 N) $\rightarrow \text{Level}(\text{CAPTAIN}) = \text{Secret}$ and Fuzzyvalue = 0.8.

Complex constraints

The examples of constraints that we have given above are enforced on a single relations only. Note that constraints can also be enforced across relations. We call such constraints complex constraints. An example is given below:

$R_1(A_1, A_2, \dots, A_n) \& R_2(B_1, B_2, \dots, B_m) \& R_1.A_i = R_2.B_j (1 < i < n, 1 < j < m) \rightarrow \text{Level}(\text{Together}(A_k, B_p)) = \text{Secret}$ where $1 < k < n, 1 < p < m$

This constraint states that pair of values involving the k th attribute of R_1 and the p th attribute of R_2 are Secret provided the corresponding values (i.e. in the same row) of the i th attribute of R_1 and the j th attribute of R_2 are equal.

1.2 APPROACH TO SECURITY CONSTRAINT PROCESSING

Security constraints enforce a classification policy. Therefore it is essential that constraints are manipulated only by an authorized individual. In our approach constraints are maintained by the SSO. That is, constraints are protected from ordinary users. We assume that constraints themselves could be classified at different security levels. However, they are stored at system-high. The constraint manager, which is trusted, will ensure that a user can read the constraints classified only at or below his level.

Our approach to security constraint processing is to handle certain constraints during query processing, certain constraints during database updates and certain constraints during database design. The first step was to decide whether a particular constraint should be processed during the query, update or database design operation. After some consideration, we felt that it was important for the query processor to have the ability to handle all of the security constraints. Our thesis is that inferences can be most effectively handled, and thus prevented during query processing. This is because most users usually build their reservoir of knowledge from responses that they receive by querying the database. It is from this reservoir of knowledge that they infer unauthorized information. Moreover, no matter how securely the database has been designed, users could eventually violate security by inference because they are continuously updating their reservoir of knowledge as the world evolves. It is not feasible to have to redesign the database simultaneously.

The next step was to decide which of the security constraints should be handled during database updates. After some consideration, we felt that except for some types of constraints such as the release and aggregate constraints, the others could be processed during the update operation. However, techniques for handling constraints during database updates could be quite complex as the security levels of the data already in the database could be affected by the data being updated.

Therefore, initially our algorithms handle only the simple and content-based constraints during database updates.

The constraints that seemed appropriate to handle during the database design operation were those that classified an attribute or collections of attributes taken together. These include the simple and association-based constraints. For example, association-based constraints classify the relationships between attributes. Such relationships are specified by the schema and therefore such constraint could be handled when the schema is specified. Since a logical constraint is a rule which specifies the implication of an attribute from a set of attributes, it can also be handled during database design.

Note that some constraints can be handled in more than one way. For example, we have the facility to handle the content-based constraints during query processing as well as during database updates. However, it may not be necessary to handle a constraint in more than one place. For example, if the content-based constraints are satisfied during the database update operation, then it may not be necessary to examine them during query processing also. Furthermore, the query operation is performed more frequently than the update operation. Therefore, it is important to minimize the operations performed by the query processor as much as possible to improve performance. However, there must be a way to handle all of the constraints during query processing. This is because, if the real-world is dynamic, then the database data may not satisfy all of the constraints that are enforced as integrity rules, or the schema may not satisfy the constraints that are processed during database design. This means that there must be a trigger which informs the query processor that the multilevel database or the schema is not consistent with the real-world; in which case the query processor can examine the additional constraints. A schematic representation of the approach to constraint generation and enforcement disclosed here is shown in FIG. 5.

2. DESIGN AND IMPLEMENTATION OF THE QUERY PROCESSOR

2.1 OVERVIEW

We first describe a security policy for handling inferences during query processing and then discuss our implementation approach.

2.1.1 SECURITY POLICY

A security policy for query processing that we propose extends the simple security property in Bell, D., and L. La Padula, July 1975, "Secure Computer Systems: Unified Exposition and Multics Interpretation," Technical Report NTIS AD-A023588, The MITRE Corporation to handle inference violations. This policy is stated below.*

Such a policy was first proposed in the LDV design [HONEY87].

1. Given a security level L , $E(L)$ is the knowledge base associated with L . That is, $E(L)$ will consist of all responses that have been released at security level L over a certain time period and the real world information at security level L .
2. Let a user U at security level L pose a query. Then the response R to the query will be released to this user if the following condition is satisfied:
For all security levels L^* where L^* dominates L , if $(E(L^*)UR) = \Rightarrow X$ (for any X) then L^* dominates $\text{Level}(X)$. Where $A = \Rightarrow B$ means B can

be inferred from A using any of the inference strategies and $\text{Level}(X)$ is the security level of X .

We assume that any response that is released into a knowledge base at level L is also released into the knowledge bases at level $L^* \geq L$. The policy states that whenever a response is released to a user at level L , it must be ensured that any user at level $L^* \geq L$ cannot infer information classified at a level $L^+ > L^*$ from the response together with the knowledge that he has already acquired. Note that while we consider only hierarchical levels in specifying the policy, it can be extended to include non-hierarchical levels also.

2.1.2 FUNCTIONALITY OF THE QUERY PROCESSOR

The strength of the query processor depends on the type of inference strategies that it can handle. Our prototype handles a limited set of inference strategies. Nevertheless it is a useful prototype which enhances the security of existing multilevel secure relational database management systems. In this section, we discuss the techniques that we have used to implement the security policy. They are: query modification and response processing. Each technique is described below.

Query modification

Query modification technique has been used in the past to handle discretionary security and views. Stonebraker, M., and E. Wong, 1974j, "Access Control in Relational Database Management Systems by Query Modification," Proceedings ACM National Conference, New York, N.Y. This technique has been extended to include mandatory security in Dwyer, P., G. Jelatis, B. Thuraisingham, Juen 1987, "Multilevel Security in Database Management Systems," *Computers and Security*, Volume 6, No. 3, pp. 252-260. In our design of the query processor, this technique is used by the inference engine to modify the query depending on the security constraints, the previous responses released, and real world information. When the modified query is posed, the response generated will not violate security.

Consider the architecture for query processing illustrated in FIG. 1. The inference engine has access to the knowledge base which includes security constraints, previously released responses, and real world information. Conceptually one can think of the database to be part of the knowledge base. We illustrate the query modification technique with examples. The actual implementation of this technique could adapt any of the proposals given in Gallaire, H., and J. Minker, 1978, *Logic and Databases*, Plenum Press for deductive query processing. Our implementation is described in section 2.2.

Consider a database which consists of relations SHIPS and ASSIGNMENT where the attributes of SHIPS are $S\#, SNAME, CAPTAIN$ and $A\#$ with $S\#$ as the key; and the attributes of ASSIGNMENT are $A\#, MISSION$ and $DESTINATION$ with $A\#$ as the key. Let the knowledge base consist of the following rules:

1. $\text{SHIPS}(X, Y, Z, D)$ and $Z = \text{Smith} \rightarrow \text{Level}(Y, \text{Secret})$
2. $\text{SHIPS}(X, Y, Z, A)$ and $A = 10 \rightarrow \text{Level}(Y, \text{Top Secret})$
3. $\text{SHIPS}(X, Y, Z, A) \rightarrow \text{Level}((Y, Z), \text{Secret})$
4. $\text{SHIPS}(X, Y, Z, A)$ and $\text{Release}(Z, \text{Unclassified}) \rightarrow \text{Level}(Y, \text{Secret})$
5. $\text{SHIPS}(X, Y, Z, A)$ and $\text{Release}(Y, \text{Unclassified}) \rightarrow \text{Level}(Z, \text{Secret})$
6. $\text{NOT}(\text{Level}(X, \text{Secret}) \text{ or } \text{Level}(X, \text{Top Secret})) \rightarrow \text{Level}(X, \text{Unclassified})$

The first rule is a content-based constraint which classifies a ship name whose captain is Smith at the Secret level. Similarly, the second rule is also a content-based constraint which classifies a ship name whose assignment number is 10 at the Top Secret level. The third rule is an association-based constraint which classifies ship names and captains taken together at the Secret level. The fourth and fifth rules are additional restrictions that are enforced as a result of the context-based constraint specified in rule 3. The sixth rule states that the default classification level of a data item is Unclassified.

Suppose an Unclassified user requests the ship names in SHIPS.

This query is represented as follows:

SHIPS(X,Y,Z,A)

Since a ship name is classified at the Secret level if either the captain is "Smith" or the captain name is already released at the Unclassified level, and it is classified at the TopSecret level if the assignment is "10", assuming that the captain names are not yet released to an Unclassified user, the query is modified to the following:

SHIPS(X,Y,Z,D) and $Z \neq \text{Smith}$ and $A \neq 10$.

Note that since query modification is performed in real-time, it will have some impact on the performance of the query processing algorithm. However, several techniques for semantic query optimization have been proposed recently for intelligent query processing in a non-secure environment (see, for example, Minker, J., "foundations of Deductive Database," Morgan Kaufman, 1988). These techniques could be adapted for query processing in a multilevel environment in order to improve the performance.

Response Processing

For many applications, in addition to query modification, some further processing of the response such as response sanitization may need to be performed. We will illustrate this point with examples.

EXAMPLE

Consider the following release constraints discussed earlier. That is,

- (i) all ship names whose corresponding captain names are already released to Unclassified users are Secret, and
- (ii) all captain names whose corresponding ship names are already released to Unclassified users are Secret.

Suppose an Unclassified user requests the ship names first. Depending on the other constraints imposed, let us assume that only certain names are released to the user. Then the ship names released have to be recorded into the knowledge base. Later, suppose an Unclassified user (does not necessarily have to be the same one) asks for captain names. The captain name values (some or all) are then assembled in the response. Before the response is released, the ship names that are already released to the Unclassified user need to be examined. Then the captain name value which corresponds to a ship name value that is already released is suppressed from the response. Note that there has to be a way of correlating the ship names with the captains. This means the primary key values (which is the S#) should also be retrieved with the captain names as well as be stored with the ship names in the release database.

EXAMPLE

Consider the following aggregate constraint

Suppose an Unclassified user requests the tuples in SHIPS. The response is assembled and then examined to see if it has more than 10 tuples. If so, it is suppressed.

There are some problems associated with maintaining the release information. As more and more relevant release information gets inserted, the knowledge base could grow at a rapid rate. Therefore efficient techniques for processing the knowledge base need to be developed. This would also have an impact on the performance of the query processing algorithms. Therefore, one solution would be to include only certain crucial release information in the knowledge base. The rest of the information can be stored with the audit data which can then be used by the SSO for analysis.

2.2. DESIGN AND IMPLEMENTATION

In section 2.2.1, we describe the various architectures that we considered for the implementation and the selected architecture. In section 2.2.2, we describe the representation of the constraints. In section 2.2.3, we describe the modules of the query processor. In section 2.2.4, we discuss some other issues concerning our prototype.

2.2.1 ARCHITECTURE COMPARISON

Alternate Architectures

We examined three different architectures for the implementation. A description of each architecture is given below.

- (i) In the first architecture, the database as well as the knowledge base is considered to be a set of Prolog clauses. Query processing would then amount to thereon proving. Many expert system have been developed using Prolog (see, for example, Merritt, D., 1989, *Building Expert Systems in Prolog*, Springer Verlag, New York). These systems take advantage of the backward chaining mechanisms provided by Prolog. In addition, several other reasoning mechanisms have also been implemented using Prolog. Implementing the query processor in Prolog, would produce a fairly powerful system.* the implementations described in [ROWE89] use such an architecture.
- (ii) The second alternative is to augment a relational database management system with a theorem prover implemented in Prolog. The advantages of augmenting a relational database system with an inference engine are discussed in Li, D., 1984, *A Prolog Database System*, Research Studies Press, John Wiley and Sons. Many commercial relational systems already have a Prolog interface.
- (iii) As the third alternative, we considered an architecture where a multilevel relational database system was augmented with an inference engine. Such an architecture would be useful as the multilevel relational database system would ensure the enforcement of a basic mandatory security policy. The inference engine then needs to implement only the policy extensions which are enforced in order to handle inferences.

After examining the three architectures, we decided to select the third one. This was because we are interested in handling security violations via inference for database systems which are already considered to be secure. Commercial multilevel relational systems are already available. Therefore, we feel that in order to produce a useful prototype we need to use such a system which will enforce the basic mandatory security policy.

Implementation Architecture

Once we had settled on the architecture, the next task was to select a multilevel relational database system for the implementation. After investigating the various systems that were available, we selected the Secure SQL Server Sybase Inc. "Secure SQL Server," 1989 for the following reasons:

- (i) for system was already available for our use,
- (ii) we had prototyping experiences with the nonsecure version of SYBASE DataServer,
- (iii) the system provided the basic security features that we needed.

A detailed discussion on Secure SQL Server is given in Rougeau, P., and E. Stearns, "The Sybase Secure Database Server: A Solution to the Multilevel Secure DBMS Problem," Proceedings of the 10th National Computer Security Conference, Baltimore, Md., 1987. Note that Secure SQL Server enables the use of sixteen security levels numbered 1 through 16. The basic mandatory security policy enforced is read at or below your level and write at your level.

A high level implementation architecture is shown in FIG. 6. In this architecture, the Secure SQL Server is augmented with an Inference Engine. We have stored the knowledge in the database. This way, the knowledge in the knowledge base can also be protected by the Secure DataServer. The Inference Engine does query modification as well as response processing.

We implemented the Inference Engine in "C" because of the C programming language interface that already exists for the Secure SQL Server. In the long-term, we envisage replacing such an Inference Engine with a more powerful logic-based theorem prover.

2.2.2 REPRESENTATION OF SECURITY CONSTRAINTS

We assume that the constraints are maintained by the SSO. Constraints themselves could be classified at different levels. However they are stored at system-high. The constraint manager, which is a trusted process, will ensure that a constraint classified at level L can only be read by a user cleared at level L or higher.

The constraints are entered in a format that is a simplified version of the rules that we described in section 1. The constraints entered by the SSO are then processed by a module of the Query processor and stored in a graphical structure. We found this an efficient way to represent the constraints. We have developed algorithms to scan the graph structure in order to obtain the relevant constraints during query processing. The algorithms also perform some optimization for efficiency. The graph structure is illustrated in FIG. 7. The relations are combined to form a linked list. Each relation has sixteen pointers emanating from it; one for each security level. Associated with each level is a lined list of constraints. Each constraint has a set of attributes that it classifies, constraint specific information such as events and conditions, and a pointer to the next constraint. Our implementation allows for the specification of events and conditions which are quite complex. Each constraint that is associated with a level classifies a set of attributes at that level.*

An alternate representation of constraint is discussed in section 5.

2.2.3 MODULES OF THE QUERY PROCESSOR

An overview of the major modules is shown in FIG. 8. The query processor consists of five modules P1 through P5. Each module is implemented as an Ultrix process.* The processes communicate with each other via the socket mechanism. A brief overview of the

functions of each module is given below. We also identify the trust that must be placed on each process. although operating system used in the implementation is not secure, our design assumes a the use of a multilevel secure operating system.

Process P1: The User Interface Manager

This process asks for password and security level from the user. Since we assume that the operating system is secure, we rely on the identification and authentication mechanism provided by the operating system. Due to this feature, P1 need not be a trusted process. It operates at the user's level. P1 accepts a query from the user and performs syntax check. It then sends the query to process P2 and returns the response received from P2 to the user. It then waits in idle state for a request from the user.

Process P2: The Central Inference Controller

This process first sets up communication with P1. It then waits in idle state for requests from P1. When a request arrives from P1, it logs into the database server at the user's level. It then requests process P3 (via socket) to return applicable constraints. The query is then modified based on the constraints (if any). The modified query is then sent to the MLS/DBMS. The response is then sent to process P4 for further processing. When P4 returns the sanitized response, a request is sent to process P5 to update the release database and the response is given to P1. P2 then returns to idle state. If constraints classified at a higher level are not processed by P3 or if the response from the MLS/DBMS is first given to P4 and P5 for sanitization and release database update, then P2 need not be a trusted processes. However, in our implementation, since P2 could have access to higher level information it must be trusted.

Process P3: The Constraint Gatherer

This process first sets up socket for communication with P2 and then logs into the database server at system-high. This is because P3 examines not only the security constraints classified at or below the user's level, but also higher level constraints. These higher level constraints are examined to ensure that by releasing a response at level L, it is not possible for users cleared at a higher level to infer information to which they are not authorized. P3 builds and maintains the constraint table whenever the constraints are updated. It waits in idle state for requests from P2. When a request arrives, it builds a list of applicable constraints and sends the constraint structure to P2 and then returns to idle state. Since P3 maintains the security constraints it is a trusted process.

Process P4: The Sanitizer

This process sets up socket for communication with P2 and logs into the database server at system high. It waits in idle state for a request to arrive from P2. When a request arrives, which consist of the response and the applicable release constraints, it sanitizes the response based on the information that has previously been released. It reads the release database maintained at various levels in order to carry out the sanitization process. It then returns the sanitized response to P2 and returns to idle state. Since response sanitization is a security critical function, P4 must be trusted.

Process P5: The Release Database Updater

This process sets up communication with P2. It waits in idle state for requests from P2. When a request arrives, it logs into the database server at all levels from system-high to the user's level and updates the release database at each level depending on the release constraints for that level. Note that this is necessary only if

higher level constraints are examined by P3. If not, P5 can log into the database server only at the user's level. After each update to the release database, it logs out of the database server at each level. It returns a status message to P2 upon completion and returns to idle state.

2.2.5 GENERAL DISCUSSION

Over 8500 lines of C code have been implemented in the development of this prototype. We first developed the infrastructure of the query processor program. This involved the creation of the five processes and establishing the necessary communications between them. Some of these processes also had to log into Secure SQL Server at the appropriate security levels. The program was set up in such a way as to leave hooks for the easy addition of more features. Since this project is a preliminary prototype of a system which could conceivably be extended in the future, we have tried to continue this approach of flexibility and modularity to make further expansion of the program easier.

It appears that there is a noticeable performance degradation when individual release constraints are handled. Large amounts of data need to be recorded. There are possibilities for optimization and this will be part of our future work. From the experiments that we have carried out so far, the performance impact of handling all of the other constraints is marginal. That is, there is hardly any visible difference between the execution times of the query processing strategy with or without the query processor.

It should be noted that in our implementation we have assumed that the process P3 examines constraints not only classified at or below the user's level, but also the higher level constraints. That is, the higher level

constraints have an impact on the query modification. From the responses received a user may be able to infer the constraints at the higher level. If on the other hand the higher level constraints are not processed by P3, the response may contain sensitive information. One way to overcome this problem is to analyze the constraints before they are enforced so that it is not possible for higher level constraints to have an impact on lower level query processing. More research needs to be done on constraint analysis.

2.3 TEST SCENARIOS

In this section we illustrate the processing of the query processor with some examples.

Let the database consist of two relations SHIPS and GROUPS. The attributes of SHIPS are Number, Name, Class, Date, and Assignment. Its primary key is Number. The attributes of GROUPS are Number, Location, Mission, and Siop. Its primary key is Number. We assume that SHIPS.Assignment and GROUPS.Number take values from the same domain. Also, SHIPS.Assignment is a foreign key. The database is populated as shown below. To simplify the discussion we assume that both SHIPS and GROUPS are assigned level 1. Furthermore, all of the tuples are also stored at level 1. Note that the usual DOD classification levels do not exist as such in the secure DBMS that we have used. We assume that the number 1 denotes the Unclassified level, the number 10 denoted the Secret level, and the number 16 (which is system-high) denoted the TopSecret level, and 10=secret, 16=top secret, i.e. system high. The user is assumed to be logged in at level 1. The table #filter temp 1 is a temporary work table used to store the result.

Relation SHIPS				
Number	Name	Class	Date	Assignment
CVN 68	Nimitz	Nimitz	May 75	003
CV 67	John F Kennedy	John F Kennedy	Sep 68	001
BB 61	Iowa	Iowa	Feb 43	003
CG 47	Ticonderoga	Ticonderoga	Jan 83	005
DD 963	Spruance	Spruance	Sep 75	006
AGF 3	La Salle	Converted Raleigh	Feb 64	003
WHEC 715	Hamilton	Hamilton	Feb 67	003
FFG 7	Oliver Hazard Perry	Oliver Hazard Perry	Dec 77	001
FF1052	Knox	Knox	Apr 69	001
LSD 36	Anchorage	Anchorage	Mar 69	009
LHA 1	Tarawa	Tarawa	May 76	003
MCM 1	Avenger	Avenger	Sep 87	003
AOR 1	Whichita	Whichita	Jun 69	003
AFS 1	Mars	Mars	Dec 63	001
AE 21	Suribachi	Suribachi	Nov 56	009
AE 23	Nitro	Nitro	May 59	005
AO 177	New Cimarron	New Cimarron	Jan 81	001
SSN 706	Albuquerque	Los Angeles	May 83	006
CVN 65	Enterprise	Enterprise	Nov 61	009
MSO 427	Constant	Aggressive	Sep 54	001

Relation Groups			
Number	Location	Mission	Siop
001	North Atlantic	naval exercises	001
002	South Atlantic	falklands patrol	002
003	Mediterranean	iraq crisis	006
004	Philippines	stabilize government	005
005	Persian Gulf	iraq crisis	004
006	Indian Ocean	naval exercises	004
007	North Sea	soviet reconnaissance	003
008	North Atlantic	oceanographic research	003
009	North Pacific	oceanographic research	001

TEST SCENARIO 1: No Constraints

Constraints active: NONE

Original query: select * from Ships

User's level: 1

Final modified query: Same as the original query (that is, query is not modified)

-continued

select ships.number, ships.name, ships.class, ships.data, ships.assignment
into #filter_temp1 from ships

Note that the asterisk is a wildcard indicator which means the query is for all attributes (fields) in a record. When the Inference Engine sees this character it replaces it with all the field names in any tables specified in the from clause.

Result: All of the tuples in SHIPS

TEST SCENARIO 2: Content constraints

Constraints active:

ships.class = 'Belknap' → Level(ships.class) = 16;
ships.class = 'Ticonderoga' → Level(ships.class) = 16;
ships.class = 'Leahy' → Level(ships.class) = 16;
ships.class = 'Charles F Adams' → Level(ships.class) = 16;
ships.class = 'Ohio' → Level(ships.class) = 16;
ships.class = 'Spruance' → Level(ships.class) = 16;
ships.class = 'Iowa' → Level(ships.class) = 16;
ships.class = 'Aggressive' → Level(ships.class) = 16;
ships.class = 'Mars' → Level(ships.class) = 16;
ships.class = 'Nimitz' → Level(ships.class) = 16;
ships.class = 'Los Angeles' → Level(ships.class) = 16;
ships.class = 'John F Kennedy' → Level(ships.class) = 16;
ships.class = 'Enterprise' → Level(ships.class) = 16;
ships.class = 'Anchorage' → Level(ships.class) = 16;

Original query: select * from ships

User's level: 1

Final modified query:

select ships.number, ships.name, ships.class, ships.date, ships.assignment
into #filter_temp1 from ships where
(not (ships.class = 'Belknap')) and
(not (ships.class = 'Ticonderoga')) and
(not (ships.class = 'Leahy')) and
(not (ships.class = 'Charles F Adams')) and
(not (ships.class = 'Ohio')) and
(not (ships.class = 'Spruance')) and
(not (ships.class = 'Iowa')) and
(not (ships.class = 'Aggressive')) and
(not (ships.class = 'Mars')) and
(not (ships.class = 'Nimitz')) and
(not (ships.class = 'Los Angeles')) and
(not (ships.class = 'John F Kennedy')) and
(not (ships.class = 'Enterprise')) and
(not (ships.class = 'Anchorage'))

Result:

Number	Name	Class	Date	Assignment
AGF 3	La Salle	Converted Raleigh	Feb 64	003
WHEC 715	Hamilton	Hamilton	Feb 67	003
FFG 7	Oliver Hazard Perry	Oliver Hazard Perry	Dec 77	001
FF1052	Knox	Knox	Apr 69	001
LHA 1	Tarawa	Tarawa	May 76	003
MCM 1	Avenger	Avenger	Sep 87	003
AOR 1	Whichita	Whichita	Jun 69	003
AE 21	Suribachi	Suribachi	Nov 56	010
AE 23	Nitro	Nitro	May 59	005
AO 177	New Cimarron	New Cimarron	Jan 81	001

TEST SCENARIO 3: Logical Constraints

Constraints active:

Logical(groups.location → groups.mission);
Level(groups.mission) = 16

Original query: select ships.name, groups.location, groups.siop from ships,
groups where ships.assignment = groups.number

Final modified query:

select ships.name, groups.siop into #filter_temp1 from ships, groups where
ships.assignment = groups.number

Results:

Name	Siop
Nimitz	006
John F Kennedy	001
Iowa	006
Ticonderoga	004
Spruance	004
La Salle	006
Hamilton	006
Oliver Hazard Perry	001
Knox	001
Anchorage	001
Tarawa	006
Avenger	006
Whichita	006
Mars	001
Suribachi	001

-continued

Nitro	004
New Cimarron	001
Albuquerque	004
Enterprise	001
Constant	001

TEST SCENARIO 4: Association Constraint (or Together Constraint)

constraints active:

Level(Together(groups.mission, groups.location)) = 10

Original query: select * from groups

User's level: 1

Final modified query:

select groups.number, groups.location, groups.siop into #filter_temp1 from groups

Results:

Number	Location	Siop
001	North Atlantic	001
002	South Atlantic	002
003	Mediterranean	006
004	Philippines	005
005	Persian Gulf	004
006	Indian Ocean	004
007	North Sea	003
008	North Atlantic	003
009	North Pacific	001

TEST SCENARIO 5: Content and Logical Constraints

Constraints Active:

Logical(Groups.Mission → Groups.Location)

Groups.Location = Persia Gulf → Level(Groups.Location) = 16

Original query: select * groups

Final modified query:

Select groups.number, groups.location, groups.mission, groups.siop into #filter_temp1 from groups where (not(groups.location = 'Persian Gulf'))

Number	Location	Mission	Siop
001	North Atlantic	naval exercises	001
002	South Atlantic	falklands patrol	002
003	Mediterranean	iraq crisis	006
004	Philippines	stabilize government	005
006	Indian Ocean	naval exercises	004
007	North Sea	soviet reconnaissance	003
008	North Atlantic	oceanographic research	003
009	North Pacific	oceanographic research	001

TEST SCENARIO 6: Release Constraint

Constraints active: Release(ships.assignment:1) → Level(ships.name) = 10

(i.e. if ships.assignment is released at level 1, then ships.name is classified at level 10)

Original query: select * from ships

User's level: 1

Results released previously were cleared before executing this query.

Release constraint triggered by the release of:

ships.assignment at level 1, ships.name can't appear in query.

Final modified query:

select ships.number, ships.class, ships.date, ships.assignment into #filter_temp1 from ships

Result:

Number	Class	Date	Assignment
CVN 68	Nimitz	May 75	003
CV 67	John F Kennedy	Sep 68	001
BB 61	Iowa	Feb 43	003
CG 47	Ticonderoga	Jan 83	005
DD 963	Spruance	Sep 75	006
AGF 3	Converted Raleigh	Feb 64	003
WHEC 715	Hamilton	Feb 67	003
FFG 7	Oliver Hazard Perry	Dec 77	001
FF1052	Knox	Apr 69	001
LSD 36	Anchorage	Mar 69	010
LHA 1	Tarawa	May 76	003
MCM 1	Avenger	Sep 87	003
AOR 1	Whichita	Jun 69	003
AFS 1	Mars	Dec 63	001
AE 21	Suribachi	Nov 56	010
AE 23	Nitro	May 59	005
AO 177	New Cimarron	Jan 81	001
SSN 706	Los Angeles	May 83	006
CVN 65	Enterprise	Nov 61	010
MSO 427	Aggressive	Sep 54	001

Release Table contents:

Name	Level
ships.number	1

2025 RELEASE UNDER E.O. 14176

-continued

ships.class	1
ships.date	1
ships.assignment	1

TEST SCENARIO 7: Aggregate ConstraintConstraints active: Aggregate(10) \rightarrow Level(ships.name) = 12;

Original query: select * from ships

User's level: 1

Final query: Same as original query

select ships.number, ships.name, ships.class, ships.date, ships.assignment
into #filter__temp1 from shipsResult: No result returned for the query since more than 10 ship names
would have been returned.**TEST SCENARIO 8: Aggregate Constraint**Constraints active: Aggregate(10) \rightarrow Level(ships.name) = 12;

Original query: select * from ships where number like '% CV %'

final query as modified

select ships.number, ships.name, ships.class, ships.date, ships.assignment
into #filter__temp1 from ships where number like '% CV %'

Result:

Number	Name	Class	Date	Assignment
CVN 68	Nimitz	Nimitz	May 75	003
CV 67	John F Kennedy	John F Kennedy	Sep 68	001
CVN 65	Enterprise	Enterprise	Nov 61	010

3 DESIGN AND IMPLEMENTATION OF THE UPDATE PROCESSOR

3.1 OVERVIEW

MLS/DBMSs ensure the assignment of a security level to data as data is inserted or modified. The security level assigned to the data, however, is generally assumed to be the login security level of the user entering the data. A more powerful and dynamic approach to assigning security levels to data is through the utilization of security constraints, or classification rules, during update operations. This section provides an overview of the functionality and utilization of a tool, the Update Processor, that utilizes security constraints as its mechanism for determining the security level of data being inserted or modified. Descriptions of the security policy and of the types of security constraints addressed by the Update Processor are also included.

3.1.1 SECURITY POLICY

The security policy of the Update Processor is formulated from the simple security property in Bell, D., and L. La Padula, July 1975, "Secure Computer Systems: Unified Exposition and Multics Interpretation," Technical Report NTIS AD-A023588, The MITRE Corporation and from a security policy provided by our underlying MLS DBMS, SYBASE's Secure SQL Server. This policy is as follows:

1. All users are granted a maximum clearance level. A user may log in at any level that is dominated by his maximum clearance level. Subjects act on behalf of users at the user's login security level.
2. Objects are the rows, tables, and databases, and every object is assigned a security level upon creation.
3. A subject has read access to an object if the security level of the subject dominates the security level of the object.
4. A subject has write access to an object if the security level of the object dominates the security level of the subject.

Statements 3 and 4 of the policy presented above are the simple and *-property of the Bell and LaPadula policy. Since the Secure SQL Server by default polyinstantiates with updates, we are utilizing the more relaxed security policy offered by the Secure SQL Server. This less strict security policy is provided via the relaxation property option. The relaxation property does

polyinstantiate with inserts, does not polyinstantiate with updates and allows users to delete tuples which their login security level dominates. More details on the security policy of the Secure SQL Server are provided in Rougeau, P., and E. Stearns, "The Sybase SEcure Database Server: A Solution to the Multilevel Secure DBMS Problem," Proceedings of the 10th National Computer Security Conference, Baltimore, Md., 1987.

3.1.2 FUNCTIONALITY OF THE UPDATE PROCESSOR

The Update Processor utilizes simple and content-dependent security constraints as guidance in determining the security level of the data being updated. The use of security constraints can thereby protect against users incorrectly labelling data as a result of logging in at the wrong level, against data being incorrectly labelled when it is imported from systems of different modes of operation such as a system high, and against database inconsistencies as a consequence of the security label of data in the database being affected by data being entered into the database.

The security level of an update request is determined by the Update Processor as follows. The simple and content-dependent security constraints associated with the relation being updated and with a security label greater than the user login security level are retrieved and examined for applicability. If multiple constraints apply, the security level is determined by the constraint that specifies the highest classification level. If no constraints apply, the update level is the login security level of the user. The Update Processor, therefore, does not determine the security level of the data solely from the security constraints, but utilizes the constraints as guidance in determining the level of the input data. The following examples illustrate the functionality of the Update Processor.

Consider a database that consists of a relation SHIPS whose attributes are number, name, class, date, and assignment, and number as its primary key. The content-based constraint which classifies all ships with name Georgia as secret is expressed as:
SHIPS.name="Georgia" \rightarrow Secret.

A user at login security level confidential enters the following data to insert a tuple into the SHIPS relation:

Insert SHIPS values ("SSBN 729", "Florida", "Ohio", "Feb 84", "008"). The Update Processor will receive this insert and retrieve the constraints associated with the SHIPS relation which specify a level greater than the user level, which is confidential, and whose level is less than or equal to the user level. The content-based constraint stated above is retrieved. Since the data entered for the name field is not "Georgia", the security constraint associated with the SHIPS relation will not affect the classification level of the insert, and the Update Processor will determine the insert level to be the user level, which is confidential.

Suppose a user at login security level confidential then enters the following: Insert SHIPS values ("SSBN 730", "Georgia", "Ohio", "Mar 89", "009"). The Update Processor will again retrieve the content-based constraint associated with the SHIPS relation, which specifies a level greater than the user level and whose level is less than or equal to the user level. Since the data for the name field is "Georgia", the Update Processor will determine the insert level to be secret. If, however, the user entered this insert at login security level top secret, the Update Processor would perform the insert at the user level since the user level is higher than the level specified by the security constraint.

The update operation of the Update Processor functions similarly to the insert operation. As an example, suppose a user at the confidential level enters the following: Update SHIPS set name="Georgia" where class="Ohio". the Update Processor will retrieve the security constraints associated with the SHIPS relation which specify a level greater than the user level and whose level is less than or equal to the user level. The content-dependent constraint stated above will be retrieved, and the Update Processor will determine the update level to be secret since the name field is being modified to "Georgia". The tuple with a primary key of "SSBN 729" as defined above will then be updated at the secret level, and the original tuple will be deleted.

In addition to describing the functionality of the Update Processor, the examples above illustrate the potential signaling channels that exist when operating with the Update Processor. A signaling channel is a form of covert channel which occurs when the actions of a high user or subject interfere with a low user or subject in a visible manner. Potential signaling channels occur when data is entered at a level higher than the user level and the user attempts to retrieve the data that he has entered, or when the Update Processor attempts to enter data at a higher level, but cannot since a tuple with the same primary key already exists at this level. We will discuss the potential signaling channels that could occur operating with the Update Processor and our solutions in Section 3.2.5.

3.1.3 UTILIZATION OF THE UPDATE PROCESSOR

An MLS DBMS provides assurance that all objects in a database have a security level associated with them and that users are allowed to access only the data which they are cleared. Additionally, it provides a mechanism for entering multilevel data but relies on the user to login at the level at which the data is to be entered. The Update Processor will provide a mechanism that can operate as a standalone tool with a MLS DBMS to provide assurance that data is accurately labelled as it is entered into the data base. This could significantly enhance and simplify the ability of an MLS DBMS to

assure that data entered via bulk data loads and bulk data updates is accurately labelled.

Another significant use for an Update Processor is in operation with an Query processor which functions during query processing. The Query processor protects against certain security violations via inference that can occur when users issue multiple requests and consequently infer unauthorized knowledge. The Query processor Prototype also utilizes security constraints as its mechanism for determining the security level of data. The security constraints are used as derivation rules as they are applied to the data during query processing. Addressing all of the security constraint types mentioned above could add a significant burden to the query processor particularly if the number of constraints is high. To enhance the performance of the query processor, the Update Processor can be utilized to address certain constraint types as data is entered into the database, in particular, simple and content-based constraints, alleviating the need for the query processor to handle these constraint types. We assume that the security constraints remain relatively static, as reliance on the Update Processor to ensure that data in the database remains consistent would be difficult, particularly in a volatile environment where the constraints change dynamically. An additional concern is that database updates could leave the database in an inconsistent state. The Update Processor, however, is designed to reject updates that cause a rippling effect and thus leave the database in an inconsistent state.

3.2 DESIGN AND IMPLEMENTATION

3.2.1 REPRESENTATION OF SECURITY CONSTRAINTS

The Update Processor handles the simple and content-based constraints. While the graph structure representation discussed in section 4 was efficient for a small number of constraints, we felt that it would be more efficient to store a large number of constraints in the database. Therefore, for the update processor prototype we decided to store the constraints in the database. As before the constraints were classified at different security levels, but stored at system-high. The owner of the constraint table is the SSO. Therefore only the SSO can manipulate the constraint table. The constraint manager, which is a trusted process, would ensure that only a user classified at level L could read the constraints classified at or below level L.

TABLE 1

CONSTRAINT Table				
C_ID	C_LEVEL	RESULT_REL_NAME	CONDITION	RESULT_LEVEL
1	6	class = "Georgia"	10	SHIPS
2	6	name = "Florida"	11	SHIPS
3	6	name = "Georgia"	12	SHIPS
4	6	1 = 1	8	SHIPS_CLASS

The CONSTRAINT table, populated with example constraints, is presented in Table 1. The definition of the field names follows. CONSTRAINT.c_id is the primary key for the table and contains a unique constraint identifier. CONSTRAINT.c_level is the constraint level. Only data entered by users with a login security level at or above this constraint level will be affected by this constraint. CONSTRAINT.result_rel_name_id is the relation name associated with the constraint. CON-

STRAINT.condition is the expression of the condition for a content-based constraint, and CONSTRAINT.result_level is the level specified by the constraint. An additional field which we recommend adding to the CONSTRAINT table is a CONSTRAINT.status field to indicate whether the constraint is currently active or inactive. The capability to change the status of constraints is particularly useful for an application whose constraints change dynamically.

3.2.2 ASSUMPTIONS

In implementing the Update Processor, the following assumptions were made. Examples are given for clarification when necessary.

1. Users can only update tuples they can see. If a user updates a tuple that exists at his login security level and the Update Processor determines the update security level to be higher than the user's login security level, the Update Processor will perform the update at this higher level. However, if a tuple with the same primary key already exists at this higher level, the update request will be rejected, as the user would in fact be updating a tuple whose security label is greater than his login security level. The Update Processor will return a request failed message to the user. We recommend that a request of this type be audited and that an SSO be alerted to resolve the conflict.

2. An update request will be aborted if it leaves the database in an inconsistent state. This may occur with the existence of more complex constraints on multiple relations. As an example: Given the constraint which references the SHIPS and SHIPS_CLASS tables, SHIPS_CLASS.length="20" → Level (SHIPS.name)=9. If the SHIPS_CLASS.length field is updated to be equal to "20," then data in the SHIPS table where SHIPS_CLASS.classification=SHIPS.class and SHIPS_CLASS.length="20" may be labelled inaccurately. An update of this type that will leave the database in an inconsistent state will be aborted.* We have designed techniques to handle such inconsistencies. These techniques have not yet been implemented.

3. If a user requests an update at a login security level that is higher than the level determined by the Update Processor, the SSO will examine the request and, if acceptable, will allow the update to be executed at the user level. The Update Processor thereby allows for the overclassification of data.

4. The Update Processor operates with the more relaxed security policy provided by SYBASE's relaxation property option. Operating with this option alleviates the need for the Update Processor to delete the original lower-level tuple when updating a tuple to a higher level since polyinstantiation is not supported with updates.

3.2.3 ALGORITHM FOR ASSIGNING SECURITY LEVELS TO DATA

Insert Request

The algorithm used by the Update Processor to determine the security level of data being inserted is as follows. Once an insert request is received, the request is parsed to retrieve the relation name. The Update Processor then searches the CONSTRAINT table for all constraints where CONSTRAINT.result_rel_name equals the relation name in the request, where the CONSTRAINT.c_level is less than or equal to the user login security level, and where the CONSTRAINT.result_level is greater than the user login security level. The applicable constraints are then ordered in descending order by CONSTRAINT.result_level. The constraints are ordered as such to alleviate the need to

examine all the constraints. The CONSTRAINT.result_level of the first constraint that applies will be the insert level determined by the constraints. Following the retrieval from the CONSTRAINT table, the initial insert request is inserted into an empty temporary table. A select statement is then built using the temporary table as the relation and the condition from the first applicable constraint as the where clause. If this select statement successfully retrieves the row in the temporary table, then the constraint applies. The CONSTRAINT.result_level for this constraint is the level at which the Update Processor will request the insert to the Secure SQL Server.

If, however, the select statement does not retrieve the row in the temporary table, the temporary table is deleted, and the algorithm repeats for the next applicable constraint. If the algorithm complete and no constraints apply, then the insert level is determined to be the user login security level.

An example of the insert algorithm is as follows. Consider the following insert requested by a user at login security level 6 on the ships database that has defined to it the constraints as specified in the CONSTRAINT table in Table 1:

insert SHIPS values ("SSBN 730", "Georgia", "Ohio", "Feb 84", "009").

Three constraints are retrieved from the CONSTRAINT table in the order CONSTRAINT.c_id="3", CONSTRAINT.c_id="2", CONSTRAINT.c_id="1". The insert request is then modified to allow the data to be inserted into an empty temporary table that has the same schema as the SHIPS table. The temporary table, #insert_temp, is created using SQL as follows: select * into #insert_temp from rel_name where 1=2, where rel_name is the relation name of the insert request. The data is then inserted into the temporary table with the following insert request: insert #insert_temp values ("SSBN730", "Georgia", "Ohio", "Feb 84", "009").

Next, a select statement is built from the CONSTRAINT.condition data for CONSTRAINT.c_id="3" and this temporary table. The select statement is:

select * from #insert_temp where name="Georgia".

Since this select statement successfully retrieves the one tuple in the temporary table, this constraint applies, and the insert level is determined to be 12, which the CONSTRAINT.result_level for this constraint. Although the other constraints may apply, the CONSTRAINT.result_level for these constraints is less than 12, so it is not necessary to examine them.

Update Request

The algorithm used by the Update Processor to determine the security level of data being updated is as follows. The request is parsed to retrieve the relation name. The Update Processor then searches the CONSTRAINT table, as it does for an insert request, for all constraints where CONSTRAINT.result_rel_name equals the relation name in the request, where the CONSTRAINT.c_level is less than or equal to the user login security level and where the CONSTRAINT.result_level is greater than the user login security level. The application constraints are then ordered in descending order by CONSTRAINT.result_level. Following the retrieval from the CONSTRAINT table, a temporary table is created with tuples from the relation in the update request that satisfy the where clause in the up-

date request. This temporary table is then utilized as it was for an insert request, i.e., as a mechanism to check if the constraints selected from the CONSTRAINT table apply. Select statements are built using the temporary table as the relation and the condition from the first applicable constraint as the where clause. If the select statement successfully retrieves any rows from the temporary table, then the constraint applies. The CONSTRAINT.result_level for this constraint is the level at which the Update Processor will request the update to the Secure SQL Server.

As with an insert request, if the select statement does not retrieve any rows in the temporary table, the temporary table is deleted, and the algorithm repeats for the next applicable constraint. If the algorithm completes and no constraints apply, the update level is determined to be the user login security level.

The following example illustrates the algorithm for update requests. Consider the update request by a user at login security level 6 on the SHIPS table which contains the tuple ("SSBN 728", "Lafayette", "Lafayette", "Jun 83", "009") and operates with constraints as defined in Table 1:

update SHIPS set name="Florida" where name="Lafayette".

Three constraints are retrieved from the CONSTRAINT table in the order CONSTRAINT.c_id="3", CONSTRAINT.c_id="2", CONSTRAINT.c_id="1".

A select statement is then built that selects into a temporary table the tuples that satisfy the condition "where name="Lafayette", which is the where clause of the update request. The select statement is: "select * into #update_temp from SHIPS where name="Lafayette" and see_label =convert(binary,user_see_label), where the test for the security label ensures that only tuples less than or equal to the user security label are selected since the process that performs this operation runs at system high. Once the temporary table is built, the update request is modified to update the temporary table. Then, the select statement is built using the where clause of the first applicable constraint as follows: "select * from #update_temp where name="Georgia"

Since this select statement does not retrieve any rows from the temporary table, the temporary table is deleted, and the algorithm repeats for the next constraint. The next constraint, SHIPS.name="Florida" = Level(SHIPS)=11, will apply, and the update level will be 11. The following subsection will describe details of the implementation design of the Update Processor.

3.2.4 MODULES OF THE UPDATE PROCESSOR

A high-level architecture for the Update Processor prototype is provided in FIG. 9. A brief description for data flow within the prototype is as follows: the User Interface* accepts a user's input and sends the input to the Secure SQL Server for a syntax check. If the syntax is correct, the user interface routes the input to the Update Processor. The Update Processor accepts the input, determines the insert/update security level for the input using the security constraints as a guideline, and establishes a connection to the Secure SQL Server at the determined security level for execution of the transaction. The Update Processor then sends a message back to the User Interface indicating the completion status of the transaction.

Minor changes were made to the user interface of the query processor for use by the Update Processor Prototype. This User Interface was

utilized to provide a common interface for the Update Processor Prototype and the Query Processor Prototype.

A more detailed presentation of the design of the Update Processor is in FIG. 10. This figure provides an overview of the modules that comprise the Update Processor. The Update Processor is modularized by function and by the security level at which the function is required to operate. Each module is implemented as an ULTRIX process, and process communication is via sockets. The underlying TCB must provide a reliable interprocess communication (IPC) mechanism for communication. A description of the functions of these modules is provided below. With this description is a discussion on the security level at which these processes run and whether they are trusted or untrusted processes.

The update Processor employs a process structure similar to the Query processor to allow for integration with the Query processor Prototype.

Process P1 provides a similar user interface to process P1 of the Query processor Prototype. Additionally, the functionality of process P2 could be integrated with the functionality of process P2 of the Query processor Prototype. However, processes P3 and P4 of the Update Processor must not be confused with processes P3 and P4 of the Query processor Prototype. Processes P3 and P4 are specific to the Update Processor. Should the Update Processor Prototype be integrated with the Query processor Prototype, processes P3 and P4 of the Update Processor must remain unique processes.

Process P1: User Interface Manager

The User Interface Manager process, process P1, provides a user interface to the Update Processor prototype. At start-up, P1 prompts the user for a password and security level. The level specified by the user is the level at which this process runs. Next, P1 prompts the user for a database request and remains in idle until it receives a request. Upon receiving a request, P1 logs into the Secure SQL Server using the user's userid, password, and clearance. The request is then sent to the server for a syntax check. The server returns a message indicating the result of the syntax check. If successful, communication is established with process P2, the Update Processor Controller, and the request, along with the login packet that contains the userid, login security level, and password of the user, is routed to P2. P1 then remains waiting for a response which will indicate the success or failure of the transaction from P2. Once a response is received from P2, P1 will display the response to the user, and P1 will again prompt the user for another request. If the user chooses to enter a request at a different login security level, process P1 will have to be restarted, at which point the user will again be prompted for a password and clearance.

The User Interface Manager operates as the front-end to the Update Processor and does not perform security-critical operations. By design, it is isolated from the operations of the higher level processes. As a result, P1 is an untrusted process and, as mentioned above, operates at the user's level.

Process P2: Update Processor Controller

The Update Processor Controller manages the flow of information between the Update Constraint Gatherer (process P3), the Level Upgrader (process P4), and the Secure SQL Server in determining the level of the update and in performing the update. Upon start-up, P2 idles, waiting for a request from P1. Upon receiving the login packet and the request, P2 logs into the Secure

SQL Server utilizing the userid, password, and clearance in the login packet. Thus, P2 runs at the user level. The Update Processor controller then examines the request to determine if it is a select, an insert, an update, or a delete. If the request is an insert or an update, some preliminary processing is performed on the request, and the request along with the login packet is sent to P3. P2 remains idle, waiting for a response which will contain the insert/update level from P3. If the level determined by P3 is greater than the user level, P2 invokes P4 to perform the insert/update level from P3. If the level determined by P3 is greater than the user level, P2 invokes P4 to perform the insert/update. P2 then idles, waiting for a successor or failure response from P4. If the level determined by P3 is the user level, then P2 sends the request to the Secure SQL Server to perform the insert/update. The Secure SQL Server returns a completion status message to P2, indicating whether the transaction completed or failed. P2 then sends this completion status message to P1 and waits for the next request from P1.

The Update Processor Controller provides assurance that the connection to the Secure SQL Server is established at the correct level, that the user's request is not modified, and that the level determined by P3 is either the level at which the update is performed or the level sent to P4. As such, the Update Processor Controller is a trusted process.

Process P3: Update Constraint Gatherer

The Update Constraint Gatherer is responsible for determining the security level of the data utilizing the applicable security constraints. Since P3 must have access to the constraints that are stored in the CONSTRAINT table, which is defined at system high, P3 runs a system high. At start-up, P3 waits for a request from P2. Upon receiving a request, P3 determines the security level of the insert or update, utilizing the algorithms described above. P3 then sends this level to P2 and idles, waiting for another request.

Since the Update Constraint Gatherer determines the level at which the insert/update will be performed, assurance must be provided that the applicable constraints are used and that the level determined by this process is accurate. This process, therefore, is a trusted process.

Process P4: Level Upgrader

The Level Upgrader is the process that issues the request to the Secure SQL Server at the level determined by P3 when the insert/update level determined by P3 is greater than the user level. (Note: P2 runs at the user level.) At start-up, P4 wait for a request from P2. Upon receiving the level from P2, P4 logs into the Secure SQL Server at this level and sends the request to the server. The response from the server is examined, and the completion status message is sent to P2. P4 then idles, awaiting another request from P2.

The Level Upgrader provides assurance that the level at which it requests the Secure SQL Server to perform the insert/update is the level received from P2. P4 is therefore a trusted process.

3.2.5 GENERAL DISCUSSION

In this section we provide a general discussion on the prototype implemented. Approximately 2500 lines of C code was implemented for the Update Processor. As mentioned earlier, the Update Processor has the ability to analyze a user's insert/update request, determine the security level of the data to be inserted/updated utilizing security constraints, and ensure that the data is in-

serted/updated at the determined level. The Update Processor can ensure that data is accurately labelled when a user enters data while logged in at the wrong level, when data is imported from systems of different modes of operation, such as a system high, or when the security level of data in the database is affected by data being entered into the database.

As discussed previously, in addition to operating as a standalone tool, the Update Processor has been designed to operate with the Query processor Prototype. As such, some of the burden placed on the Query processor can be alleviated since the simple and content-based constraints can be addressed by the Update Processor. Operating in an environment where users both query and update the database, however, allows for the occurrence of potential signaling channels. As an example, in some cases the user cannot retrieve the data he has entered. Since the security levels of the security constraints that determined the security level of the input is not at a level higher than the user level, i.e., the value of CONSTRAINT.c level for constraints used during update processing is the user level, we do not regard this as a signaling channel. The data in the CONSTRAINT table is labelled at system high to allow an SSO to maintain the table,* but the CONSTRAINT.c level value reflects the true level of the constraint. Therefore, if the constraint level, which is the value of CONSTRAINT.c level, is at or below the user level, we assume it is not the action of a high-level user or subject that is interfering with the result. SYBASE requires an SSO to be logged in at system high to have access to SSO functions.

Another significant consideration with the Update Processor operating with the Query processor Prototype is the content of error messages. The content of some of the Secure SQL Server's error messages, coupled with the ability to query the database, may enable a user to infer something about the security level of his insert/update. As an example, if it is determined that an insert request should be processed at a higher level, and if a tuple with the same primary key already exists at the higher level, then a message that indicates that a duplicate key row already exists is sent by the Secure SQL Server to the Level Upgrader. If this message were routed to the User Interface Manager, the user could infer that the data he entered exists at a higher level. Furthermore, he could infer that the data exists at a higher level either because it was input by a user at a higher level or because a security constraint exists that determined the insert level to be so. Through experimenting with additional inserts, the user could determine the existence of this security constraint. Our solution is to have the Level Upgrader interpret this error message to be a request failed message. A request failed message is then sent to the Update Processor controller, who in turn sends it to the User Interface Manager that displays it to the user. The user, therefore, is only aware that the request failed. To further resolve this confusion for the user, we recommend that transactions of this type be audited and that the SSO be alerted to provide an explanation to the user if needed.

Performance is an additional concern with the Update Processor. The response time of the Query processor may improve with the use of the Update Processor, but the response time for updates will be affected. This, however, is acceptable for an application whose percentage of retrievals exceeds that of updates. Additionally, should this functionality be incorporated into MLS

DBMS, the effect on performance may not be significant since this functionality could exist as part of the DBMS kernel rather than as a user application as it currently exists. Regardless, we project that since the performance of updates, in general, is not quite as critical as the performance of retrievals, the benefits from implementing this security functionality should outweigh the projected minimal loss in performance.

In general, the Update Processor provides functionality which is desirable in a multilevel operating environment. The nature of the tool allows for it to operate as a standalone tool or in conjunction with a Query processor. Additionally, this functionality could easily be adapted to operate with an existing MLD DBMS to enhance its security features.

3.3 TEST SCENARIOS

This section illustrates the functionality of the Update Processor. Included in this section is a description of

our test database and our test scenarios. With each test scenario is a statement of input, the significance of the test, and the results of the test. Each scenario uses our test database, the ships database, and each scenario is to be executed by a user at login security level 6.

3.3.1 TEST DATABASE

Our test database is the ships database. The ships database and all the relations in this database have been defined at level 1. The test database will initially be empty and will not be reinitialized with each scenario so the reader can see the results of utilizing the Update Processor as each transaction completes, as this is how it would operationally be used.

All of our example transactions are against the SHIPS and SHIPS CLASS relations which have been defined below.

```
create table SHIPS
(number varchar(10),
name varchar(22),
class varchar(22),
date varchar(8),
assignment varchar(10))
unique index: number
create table SHIPS CLASS
(classification varchar(50),
length varchar(50),
```

```
disp varchar(7),
speed varchar(4),
missile varchar(15),
torpedo varchar(15),
gun varchar(15))
unique index: classification
```

As mentioned in section 3.3, each user database must contain a CONSTRAINTS table to store the simple

and content-based security constraints used by the Update Processor. The following four constraints are active for these tests.

1. SHIPS.class="Ohio"→Level (SHIPS)=10;
2. SHIPS.name="Florida"→Level (SHIPS)=11;
3. SHIPS.name="Georgia"→Level (SHIPS)=12;
4. →Level (SHIPS_CLASS)=8;

3.3.2 TEST SCENARIOS

TEST SCENARIO 1: insert SHIPS values ("SSBN 728", "Lafayette", "Lafayette", "Jun 83", "009")

This scenario exemplifies an insert transaction that is not affected by the security constraints, since the value for SHIPS.class is not "ohio", and SHIPS.name is not "Florida" or "Georgia". The following response to the SQL select statement demonstrates the results of this transaction.

```
1>select sec_label, *from SHIPS
2>go
```

sec_label	number	name	class	date	assignment
0x06000000000000000000000000000000	SSBN 28	Lafayette	Lafayette	Jun 83	009

The results indicate that the tuple was not affected by the security constraints and was inserted at the user level which is level 6.

TEST SCENARIO 2:

Insert SHIPS values("SSBN 729", "Florida", "Lafayette", "Jun 83", "09")

This scenario exemplifies an insert transaction that is affected by security constraint 2. The Update Processor actually retrieves the three security constraints associated with the SHIPS relation and examines them in descending order by constraint security level. As a result, security constraint 3 is examined, followed by security constraint 2. The following retrieval demonstrates the results of this transaction.

```
1>select sec_label, *from SHIPS
2>go
```

sec_label	number	name	class	date	assignment
0x06000000000000000000000000000000	SSBN 728	Lafayette	Lafayette	Jun 83	009

This result does not indicate that the tuple was inserted. However, the Update Processor returned a successful response to the user. The user, therefore, will assume that either his tuple has since been deleted or that it was inserted at a level higher than his login security level. The following response, submitted by a user at login level 16, demonstrates the results of this transaction.

```
1>select sec_label, *from SHIPS
2>go
```

sec_label	number	name	class	date	assignment
0x06000000000000000000000000000000	SSBN 728	Lafayette	Lafayette	Jun 83	009
0xb0000000000000000000000000000000	SSBN 729	Florida	Lafayette	Jun 83	009

This response indicates that the tuple was inserted at level 11 since security constraint 2 was satisfied.

TEST SCENARIO 3:

Insert SHIPS values("SSBN 730", "Georgia", "Ohio", "Feb 84", "009")

This scenario exemplifies an insert transaction that is affected by more than one security constraint. The Up-

date Processor retrieves the three security constraints associated with the SHIPS relation in descending order by constraint security level. However, the condition from security constraint 3 is satisfied, so that the insert level is determined to be 12, and the remaining constraints are not examined. The following response, submitted by a user at login level 16, demonstrates the results of this transaction. A user logged in at level 6 would retrieve only the tuple with number="SSBN 728".

```
1>select sec_label, *from SHIPS
2>go
```

sec_label	number	name	class	date	assignment
0x060000000000000000000000	SSBN 728	Lafayette	Lafayette	Jun 83	009
0x060000000000000000000000	SSBN 729	Florida	Lafayette	Jun 83	009
0x0c0000000000000000000000	SSBN 730	Georgia	Ohio	Jun 85	006

TEST SCENARIO 4:

Update SHIPS set name="Florida" where name="Lafayette"

tion failed. It would then be the responsibility of the SSO to resolve this situation with the user.

TEST SCENARIO 6:

Insert SHIPS values("SSBN 728", "Lafayette", "Lafayette", "Jun 83", "009")

This scenario exemplifies an insert transaction that will result in a tuple being inserted at the user level, level 6, followed by an update operation that will be aborted, since it will result in a duplicate key row. The following tuples reside in the database after the above transaction is executed. The following response, submitted by a user at login level 16, demonstrates the results

of this transaction.

```
1>select sec_label, *from SHIPS
2>go
```

sec_label	number	name	class	date	assignment
0x060000000000000000000000	SSBN 729	Florida	Lafayette	Jun 83	009
0x0c0000000000000000000000	SSBN 730	Georgia	Ohio	Jun 85	006
0x060000000000000000000000	SSBN 728	Florida	Lafayette	Jun 83	009
0x060000000000000000000000	SSBN 728	Lafayette	Lafayette	Jun 83	009

This scenario exemplifies an update transaction that is affected by a security constraint. Without utilizing the Update Processor, this transaction would result in the modification to the name field in the tuple with number="SSBN 728". This tuple exists at level 6 which is the user's login security level. Utilizing the Update Processor, however, the name field will be modified as well as the classification level. Since the name is being set for "Florida", security constraint 2 will determine the update security level to be 11. The tuple will then be inserted at level 11, and the original tuple will be deleted, as we are running with the relaxation property on. The following response, submitted by a user at login level 16, demonstrates the results of this transaction. A user logged in at level 6 would not have access to the data currently in the SHIPS table.

```
1>select sec_label, *from SHIPS
2>go
```

sec_label	number	name	class	date	assignment
0x060000000000000000000000	SSBN 729	Florida	Lafayette	Jun 83	009
0x0c0000000000000000000000	SSBN 730	Georgia	Ohio	Jun 85	006
0x060000000000000000000000	SSBN 728	Florida	Lafayette	Jun 83	009

TEST SCENARIO 5:

Insert SHIPS values("SSBN 729", "Florida", "Lafayette", "June 83", "09")

This scenario exemplifies an insert transaction that will result in a duplicate key row. When a user logged in at level 6 queries the database, he will not retrieve the duplicate tuple at level 11. When this tuple is inserted, the Update Processor will determine that the insert level is level 11, will attempt to insert it at level 11, and will receive a message from the Secure SQL Server that a duplicate key row already exists. The Update Processor will then send a message to the user that the transac-

Following, the update transaction:

Update SHIPS set name="Florida" where name="Lafayette" is executed. The Update Processor will determine the update level to be 11, since security constraint 2 is satisfied, and will attempt to execute this update at level 11. The server will abort this request and return a message indicating that a duplicate key row already exists. The Update Processor will then send a message to the user that the transaction failed. It would then be the responsibility of the SSO to resolve this situation with the user.

TEST SCENARIO 7:

Insert SHIPS_CLASS values("Ohio", "Nuclear", "17", "187", "20", "Tri I", "Mk 68")

This scenario exemplifies an insert transaction that is affected by security constraint 4. Security constraint 4 is a simple constraint that specifies that all data in the relation SHIPS_CLASS will be at level 8. Therefore,

the Update Processor will determine the insert level to be level 8. The following response demonstrates the results of this transaction.

```
1>select sec_label, *from SHIPS
2>go
```

sec_label	name	classification	length	disp
0x0c0000000000000000000000	Ohio	nuclear	17	
(Attributes Continued)				
	speed	missile	torpedo	gun
	187	20	Tri I	Mk68

4. HANDLING SECURITY CONSTRAINTS DURING DATABASE DESIGN

4.1 OVERVIEW

The main focus of this section is a discussion on how association-based constraints (also called together or context-based constraints) could be handled during database design. We then briefly discuss how simple constraints as well as logical constraints could be handled.

An association-based constraint classifies a collection of attributes taken together at a particular security level. What is interesting about the association-based constraint is that it can generate several relationships between the various attributes. For example, if there is a relation SHIPS whose attributes are S#, SNAME, and CAPTAIN, and if an association-based constraint classifies the SNAME and CAPTAIN taken together at the Secret level, then one of the pairs (S#, SNAME), (S#, CAPTAIN) should also be classified at the Secret level. Otherwise, an Unclassified user can obtain the (S#, SNAME) and the (S#, CAPTAIN) pairs and infer the Secret association (SNAME, CAPTAIN). There has been much discussion in the literature as to the appropriate place to handle these association-based constraints. Some argue that they should be handled during database design Lunt, T., May 1989, "Inference and Aggregation, Facts and Fallacies," Proceedings of the IEEE Symposium on Security and Privacy, Oakland, Calif. while others argue that they should be handled during query and update processing Stachour, P., and B. Thuraishingham, June 1990, "Design of LDV—a Multilevel Secure Relational Database Management System," IEE Transactions on Knowledge and Data Engineering, Volume 2, No. 2. However, none of the work reported so far studied the properties of these association-based constraints, nor has it provided any technique to generate the additional association-based constraints that can be deduced from an initial set of association-based constraints.

We first describe an algorithm which processes a given set of association-based constraints and outputs the schema for the multilevel database. Given a set of association-based constraints and an initial schema, the algorithm will output clusters of attributes and the security level of each cluster. We then prove that the attributes within a cluster can be stored securely at the corresponding level. A tool based on this algorithm can help the systems security officer (SSO) design the multilevel database. The algorithm that we have designed does not necessarily have to be executed during database design only. It can also be executed during query processing. That is, the query processor can examine the attributes in the various clusters generated by the algorithm and then determine which information has to be released to the users. For example, if the algorithm places the attribute A1, A2 in cluster 1 at level L, and the attributes A3, A4 in cluster 2 at level L, then, after an attribute in cluster 1 has been released to a user at level L, none of the attributes in cluster 2 can be released to users at level L.

Since simple constraints can be regarded as a special form of association-based constraints, where only one attribute is classified, we feel that such constraints could also be handled during database design. Another constraint that could be handled during database design is the logical constraint. For example, if attribute A implies an attribute B, and if attribute B is classified at the Secret level, then attribute A must be classified at least

at the Secret level. It should be noted that if any of the constraints have conditions attached to them, then handling them during database design time would be difficult. For example, consider the following constraint: "Name and Destination taken together are Secret if destination is a Middle-east country". Such a constraint depends on data values. Therefore, they are best handled during either query and update processing.

The organization of this paper is as follows. In section 6.2 we describe an algorithm which determines the security levels of the attributes given a set of association-based constraints. A tool could be developed based on this algorithm which the SSO could use to design the schema. In section 6.3 we describe how simple constraints could be handled during database design. Finally in section 6.4 we discuss how logical constraint may be processed.

4.2 HANDLING ASSOCIATION-BASED CONSTRAINTS

In this section we describe an algorithm for handling association-based constraints. The input to this algorithm is a set of association-based constraints and a set of attributes. The output of this algorithm is a set of clusters for each security level. Each cluster for a security level L will have a collection of attributes that can be safely classified at the level L. That is, if A1, A2, and A4 are attributes in a cluster C at level Secret, then the attributes A1, A2, and A3 can be classified together safely at the security level Secret without violating security. The clusters are formed depending on the association-based constraints which are input to the program. Once the clusters are formed, then the database can be defined according to the functional and multivalued dependencies that are enforced.

ALGORITHM HABC (Handling Association-Based Constraints) Begin

Let C be the set of security constraints and W1, W2, Wm be the set of attributes which are input to the program.

For each security level L, do the following:

Begin

Let C[L] be the largest subset of C and A = {A1, A2, . . . An} be the largest subset of {W1, W2, . . . Wm}, such that the elements of subset of C and A are all visible at level L.

Since n is the number of attributes which are visible at level L, clusters C1, C2, . . . Cn will be formed as follows:

Set C1 = C2 = C3 = . . . = Cn = Empty-set. For each i (1 < i < n) do the following:

Begin

Find the first cluster Cj (1 < j < n) such that Ai, together with any of the attributes already in Cj, is classified at a level dominated by L by the set of constraints C[L]. Place Ai in the cluster Cj. (Note that since we have defined n clusters, there will definitely be one such Cj.)

End (for each i).

Output all the non-empty clusters along with the security level L. End (for each security level L).

End (HABL)

Theorem 1: Algorithm HABL is Sound.

Proof of Theorem 1:

We need to show that for every security level L, the attributes in a cluster formed at L can safely be stored together in a file at level L.

Let C be a cluster at level L, and let B1, B2, . . . Br be the attributes in C. Note that before each Bi is placed,

it will be first checked to determine whether or not there is an association-based constraint which classifies B_i together with any subset of the attributes B_1, B_2, \dots, B_{i-1} already in C at a level not dominated by L . If so, B_i would not have been placed in the cluster C .

Since this is true for each B_i ($1 < i < r$), there is no association-based constraint which classifies any subset of B_1, B_2, \dots, B_r taken together at a level not dominated by L . Therefore, B_1, B_2, \dots, B_r can be safely stored in a file at level L .

Theorem 2: Algorithm HABL is Complete.

Proof of Theorem 2:

We need to show that, if C_i and C_j are two clusters at a level L , there are subsets A and B , respectively, of C_i and C_j , such that A and B cannot be stored together in a file at level L .

Let $i < j$. Then the cluster C_i appears before C_j , in the enumeration of the clusters formed at level L .

Suppose, on the contrary, that A and B do not exist. Consider an element X of cluster C_j . Since C_i is before C_j in the enumeration, before placing X in C_j , it would have been first checked to determine whether or not X can be placed in C_i . It would have been found that there was an association-based constraint which classifies X together with the attributes already in subset P of C_i at a level not dominated by L . That is, the subset P and $\{X\}$ of C_i and C_j respectively cannot be stored in a file at level L . That is, we have found two sets A and B , which are subsets of C_i and C_j , respectively, which cannot be stored in a file at level L . This is a contradiction to our assumption.

We now trace the algorithm with a simple example.

Let the attributes be A_1, A_2, A_3, A_4, A_5 . Let the constraints be the following:

CON1: $A_1 \cdot A_2 = \text{Secret}^*$

By " $A_1 \cdot A_2 = \text{Secret}$ " is meant: A_1 and A_2 taken together are classified at the Secret level

CON2: $A_1 \cdot A_5 = \text{Secret}$

CON3: $A_1 \cdot A_4 \cdot A_5 = \text{Secret}$

CON4: $A_2 \cdot A_4 = \text{Secret}$

CON5: $A_3 \cdot A_4 = \text{Secret}$

Note that some of the constraints are redundant. For example, CON2 implies CON3. In this paper we are not concerned with the redundancy of the constraints.

Since the maximum classification level assigned is Secret, all the attributes can be stored in a file at the level Secret or Higher. At the Unclassified level, the following clusters are created:

$C_1 = \{A_1, A_3\}$

$C_2 = \{A_2, A_5\}$

$C_3 = \{A_4\}$

It should be noted that, although the algorithm guarantees that the constraints are processed securely, it does not provide any guarantee that the attributes are not overclassified. More research needs to be done in order to develop an algorithm which does not overclassify an attribute more than is necessary.

4.3 A NOTE ON SIMPLE CONSTRAINTS

Since simple constraints classify individual attributes at a certain security level, they could also be handled during database design. Note that when an attribute A in relation R is classified at level L , then all elements which belong to A is also classified at level L . Therefore, we can store A itself at level L .

The algorithm which handles simple constraint is straightforward. Each attribute that is classified by a simple constraint is stored at level specified in the constraint. Once the algorithm for processing simple constraints is applied and the corresponding schema is ob-

tained, then this schema is given as input to the algorithm handling association-based constraints. The association-based constraints are then applied and the final schema is obtained.

We illustrate the combined algorithm with an example. Let relation R have attributes A_1, A_2, A_3, A_4 . Let the constraints enforced be the following:

Simple constraint: A_4 is Secret Association-based constraint: A_2 and A_3 together are TopSecret.

Applying the algorithm for handling simple constraints we obtain the following A_1, A_2, A_3 are Unclassified; A_1, A_2, A_3 , and A_4 are Secret.

Next we apply the algorithm for handling association-based constraints. The final output is: A_1 and A_2 are Unclassified; A_1, A_2 , and A_4 are Secret; A_1, A_2, A_3 , and A_4 are TopSecret.

4.3 HANDLING LOGICAL CONSTRAINTS

Logical constraints are rules that can be used to deduce new data from existing data. If a security constraint classifies the new data at a level that is higher than of the existing data, then the existing data must be re-classified. Logical constraints could be straightforward such as $A_i = \Rightarrow A_j$ or they could be more complex such as $A_1 \& A_2 \& A_3 \& \dots A_n = \Rightarrow A_m$. If A_j is classified at the Secret level then A must be classified at least at the Secret level. If A_m is classified at the Secret level, then at least one of $A_1, A_2, \dots A_n$ must be classified at least the Secret level.

In section 4 we showed how the logical constraints may be handled during query processing. For example consider the constraint $A_i A_n = \Rightarrow A_j$. If A_j is classified at the Secret level, and an Unclassified user requests for A_i values, the query processor will ensure that the A_i values are not released. That is, although A_i may be explicitly assigned the Unclassified level, since the logical constraint is treated as a derivation rule, it does not cause any inconsistency. That is, during query processing, the security level of A_i will be computed to be Secret.

For logical constraints which do not have any conditions attached, it appears that they could be handled during database design. That is during design time the logical constraints are examined, and the security levels of the attributes specified in the premise of a constraint could be computed. For example, if A_j is classified at the Secret level then it must be ensured during design time that A_i classified at least at the Secret level also. The following algorithm will ensure that the security levels are computed correctly.

1. Do the following each logical constraint. (Note that we have assumed that the constraints are expressed as horn clauses. That is, there is only one atom in the head of a clause).
2. Check whether there are any simple constraints which classify the attribute appearing in the head of the logical constraint at any level. If not, ignore the constraint.
3. If so, find the highest security level L that is specified for this attribute.
4. Check whether any of the attributes appearing as premises of the logical constraint are classified at least at level L . If so, ignore the constraint.
5. If not, classify one of the attributes (say, the first one) at the level L .

The algorithm given above does not ensure that the attributes are not overclassified. In order to avoid the overclassification problem, modification must be made

to step 5. That is, once an attribute is assigned a security level, it is possible for the level to be re-assigned based on other logical constraints that are handled. Our current research includes investigating techniques for successfully assigning security levels to the attributes and at the same time avoiding overclassification.

When logical, simple, and association-based constraints are combined, then the first step would be to handle the simple constraints. The next step would be to apply the algorithm given above for the logical constraints. Finally, the algorithm given in section 6.2 is applied for the association-based constraints.

5. TOWARDS AN INTEGRATED APPROACH

Consider the integrated architecture illustrated in FIG. 5. This architecture provides an integrated solution to constraint processing in a multilevel environment. In this section we describe this integrated solution using an operational example. The example is explained in natural language.

Time to: Constraint Generator produces the following output: The database consisting of the relations SHIPS and GROUPS. Both relations are Unclassified. The attributes of SHIPS are Number, Name, Class,

domain. Also, SHIPS.Assignment is a foreign key. Following are the Security Constraints: GROUPS.Location and GROUPS.Mission taken together are Secret; SHIPS.Name is Secret if SHIPS.Class=Los Angeles.

Time T1: Database Design Tool Produces the following output: The database consists of three relations SHIPS, GROUPS, and GRP-MISS. The relations SHIPS and GROUPS are Unclassified. The relation GRP-MISS is Secret. The attributes of SHIPS are Number, Name, Class, Date, and Assignment. Its primary key is Number. The attributes of GROUPS are Number, Location and Siop. Its primary key is Number. The attributes of GRP-MISS are Number and Mission. SHIPS. Assignment and GROUPS. Number take values from the same domain. GROUPS. Number and GRP-MISS. Number take values from the same domain. Also, SHIPS. Assignment is a foreign key. Following Security Constraint is enforced: SHIPS. Name is Secret if SHIPS. Class=Los Angeles.

Time T2: The Update Processor populates the database as follows. Note that we have included security level as a field in the relations; 1=Unclassified, 10=Secret.

Relation SHIPS						
Number	Name	Class	Date	Assignment	Level	
CVN 68	Nimitz	Nimitz	May 75	003	1	
CV 67	John F. Kennedy	John F. Kennedy	Sep 68	001	1	
BB 61	Iowa	Iowa	Feb 43	003	1	
CG 47	Ticonderoga	Ticonderoga	Jan 83	005	1	
DD 963	Spruance	Spruance	Sep 75	006	1	
AGF 3	La Salle	Converted Raleigh	Feb 64	003	1	
WHEC 715	Hamilton	Hamilton	Feb 67	003	1	
FFG 7	Oliver Hazard Perry	Oliver Hazard Perry	Dec 77	001	1	
FF1052	Knox	Knox	Apr 69	001	1	
LSD 36	Anchorage	Anchorage	Mar 69	009	1	
LHA 1	Tarawa	Tarawa	May 76	003	1	
MCM 1	Avenger	Avenger	Sep 87	003	1	
AOR 1	Whichita	Whichita	Jun 69	003	1	
AFS 1	Mars	Mars	Dec 63	001	1	
AE 21	Suribachi	Suribachi	Nov 56	009	1	
AE 23	Nitro	Nitro	May 59	005	1	
AO 177	New Cimarron	New Cimarron	Jan 81	001	1	
SSN 706	Albuquerque	Los Angeles	May 83	006	10	
CVN 65	Enterprise	Enterprise	Nov 61	009	1	
MSO 427	Constant	Aggressive	Sep 54	001	1	

Relation Groups		
Number	Location	Siop
001	North Atlantic	001
002	South Atlantic	002
003	Mediterranean	006
004	Philippines	005
005	Persian Gulf	004
006	Indian Ocean	004
007	North Sea	003
008	North Atlantic	003
009	North Pacific	001

Relation GRP-MISS		
Number	Mission	Level
001	naval exercises	10
002	falklands patrol	10
003	iraq crisis	10
004	stabilize government	10
005	iraq crisis	10
006	naval exercises	10
007	soviet reconnaissance	10
008	oceanographic research	10
009	oceanographic research	10

Data, and Assignment. Its primary key is Number. The attributes of GROUPS are Number, Location, Mission, and Siop. Its primary key is Number. SHIPS.Assignment and GROUPS.Number take values from the same

Time T3: Unclassified user poses queries to select all from SHIPS and GROUPS.

Result: He will get all of the tuples in GROUPS and all of the tuples in SHIPS except the one at the level 10 (i.e. the tuple whose ship class is Los Angeles).

Time T4: Real world changes. The constraint which originally classifies locations and missions together at the Secret level is removed. Note that such a constraint is not in the constraint database. Two new constraints are introduced. One classifies tuples in GROUPS where the location is Persian Gulf at the Secret level. The other classifies ship names and assignments taken together at the Secret level.

The Constraint database is updated to include the two new constraints. We assume that no changes are made to the database or to the schema. The constraint updater informs the query processor of the inconsistency.

Time T5: An Unclassified user poses the same query; that is to retrieve all tuples from SHIPS and GROUPS.

Result: He will get all values for the attributes SHIPS. Number, SHIP. Name, SHIP. Class, and SHIPS. Date provided SHIPS. class does not have the value Los Angeles. He will not get any values for SHIPS. Assignment. He will get all tuples from GROUPS where the location is not Persian Gulf. Al-

though GRP-MISS need not be classified at the Secret level, the user will still not get any data from GRP-MISS as the relation has not yet been downgraded.

Time T6: The database is re-designed and the database data is re-classified.

There are three relations SHIPS, SH-ASSIG, and GROUPS. SHIPS and GROUPS are Unclassified, SH-ASSIG is Secret. The attributes of SHIPS are Number, Name, Class, and Date. Its primary key is Number. SH-ASSIG has attributes Number and Assignment. SH-ASSIG. Number SHIPS. Number is primary key. SHIPS. Number and SH-ASSIG. Number take values from the same domain. The attributes of GROUPS are Number, Location, Mission, and Siop. Its primary key is Number. SH-ASSIG. Assignment and GROUPS. Number take values from the same domain. Also, SH-ASSIG. Assignment is a foreign key. Following are the Security Constraints: SHIPS. Name is Secret if SHIPS. Class=Los Angeles, Each of GROUPS. Number, GROUPS. Location, GROUPS. Mission, and GROUPS. Siop is Secret if GROUPS. Location=Persian Gulf. The database is populated as shown below.

Relation Groups				
Number	Location	Mission	Siop	Level
001	North Atlantic	naval exercises	001	1
002	South Atlantic	falklands patrol	002	1
003	Mediterranean	iraq crisis	006	1
004	Philippines	stabilize government	005	1
005	Persian Gulf	iraq crisis	004	10
006	Indian Ocean	naval exercises	004	1
007	North Sea	soviet reconnaissance	003	1
008	North Atlantic	oceanographic research	003	1
009	North Pacific	oceanographic research	001	1

Relation SH-ASSIG		
Number	Assignment	Level
CVN 68	003	10
CV 67	001	10
BB 61		10
CG 47	005	10
DD 963	006	10
AGF 3	003	10
WHEC 715	003	10
FFG 7	001	10
FF1052	001	10
LSD 36	009	10
LHA 1	003	10
MCM 1	003	10
AOR 1	003	10
AFS 1	001	10
AE 1	009	10
AE 23	003	10
AO 177	001	10
SSN 706	006	10
CVN 65	009	10
MSO 427	001	10

Relation SHIPS				
Number	Name	Class	Date	Level
CVN 68	Nimitz	Nimitz	May 75	1
CV 67	John F. Kennedy	John F. Kennedy	Sep 68	1
BB 61	Iowa	Iowa	Feb 43	1
CG 47	Ticonderoga	Ticonderoga	Jan 83	1
DD 963	Spruance	Spruance	Sep 75	1
AGF 3	La Salle	Converted Raleigh	Feb 64	1
WHEC 715	Hamilton	Hamilton	Feb 67	1
FFG 7	Oliver Hazard Perry	Oliver Hazard Perry	Dec 77	1
FF1052	Knox	Knox	Apr 69	1
LSD 36	Anchorage	Anchorage	Mar 69	1
LHA 1	Tarawa	Tarawa	May 76	1
MCM 1	Avenger	Avenger	Sep 87	1
AOR 1	Whichita	Whichita	Jun 69	1
AFS 1	Mars	Mars	Dec 63	1

-continued

AE 21	Suribachi	Suribachi	Nov 56	1
AE 23	Nitro	Nitro	May 59	1
AO 177	New Cimarron	New Cimarron	Jan 81	1
SSN 706	Albuquerque	Albuquerque	May 83	10
CVN 65	Enterprise	Enterprise	Nov 61	1
MSO 427	Constant	Aggressive	Sep 54	1

Time T7: Unclassified user poses queries to select all from SHIPS and GROUPS.

He will get all tuples from SHIPS except the one with class=Los Angeles. he will get all tuples from GROUPS except the one with location=Persian Gulf. He will not get any tuples from SH-ASSIG.

We claim:

1. Apparatus for an integrated architecture for an extended multilevel secure database management system which processes security constraints to control unauthorized inference through logical deduction upon queries by users and implemented when the database is queried through the database management system, when the database is updated through the database management system, and when the database is designed, the integrated architecture comprising:

a knowledge base for storing the security constraints, application data and information on responses released from the multilevel secure database management system;

a multilevel database which contains data classified at different security levels;

a multilevel metadatabase to store schemes describing data in the multilevel database, the schemas classified at said different security levels;

the multilevel secure database management system utilized to access the multilevel database for queries and updates and to access the multilevel metadatabase for querying and updating the schemas by users cleared to said different security levels;

a query processor augmenting the multilevel secure database management system and accessing the knowledge base to examine the security constraints, application data and responses already released and to modify queries to prevent unauthorized inferences and to output a modified query for evaluation by the multilevel secure database management system, the multilevel secure database management system providing an output to the query processor which examines the security constraints, the application data, and responses already released, and modifies the responses to prevent unauthorized inferences,

an update processor augmenting the multilevel secure database management system for examining some of said security constraints and to assign security levels to the data;

the update processor complementing functions of the query processor such that if some of the constraints are processed during updates and the data is assigned appropriate security levels, said constraints need not be processed by the query processor, for performance enhancement the said update processor also being used as an off-line tool to determine the security levels of the data;

a multilevel database design tool which examines some of the security constraints and assigns security levels to the schemas, the schemas then being input to the multilevel secure database management system for storage in the multilevel metadatabase.

base at the appropriate security levels, the design tool thereby complementing the functions of the query processor so that said some of the constraints need not be processed by the query processor for performance enhancement; and

a user interface which accepts query requests from the user and passes the query to the query processor and accepts update requests from the user and passes it to the update processor if operating on-line or the user interface accepts the request from the user and passes it to the multilevel database management system if it is off-line, the user interface accepting the schema query request from the user and passes the query request to the multilevel secure database management system, the user interface further accepting the schema update requests from the user and passes it to the multilevel secure database management system.

2. The apparatus of claim 1 wherein the query processor comprises:

a user interface manager to provide the user interface and to accept the query requests;

a constraint manager to manage the security constraints and the knowledge base;

a query modifier which receives query requests from the user interface manager, examines the security constraints by communicating with the constraint manager and subsequently modifies the query which is evaluated against the multilevel secure database management system;

a response processor which accepts the response from the multilevel secure database management system, examines the security constraints by communicating with the constraint manager, and determines which parts of the response are to be released to the user; and

a release database manager which manages the release information and provides input to the query modifier and the response processor to carry out their functions.

3. The apparatus of claim 1 wherein the update processor comprises:

a user interface manager for communicating with the user;

a constraint manager which manages the security constraints;

a security level computer which communicates with the constraint manager and computes the security level of data to be updated; and

a level upgrader which gets an input from the security level computer, creates an update process at the appropriate level and interfaces to the multilevel secure database management system.

4. The apparatus of claim 1 wherein the multilevel database design tool is a monolithic module whose inputs are the security constraints and initial schemas, and whose outputs are modified schemas and their security levels which can be entered into the multilevel metadatabase.

* * * * *

[54] ASSOCIATIVE MEMORY SYSTEMS

[75] Inventor: Robert W. A. Scarr, Stansted, United Kingdom

[73] Assignee: Standard Telephones and Cables Public Limited Co., London, England

[21] Appl. No.: 747,717

[22] Filed: Jun. 24, 1985

[30] Foreign Application Priority Data

Jul. 5, 1984 [GB] United Kingdom 8417187

[51] Int. Cl.⁴ G11C 13/00[52] U.S. Cl. 365/49; 365/126;
365/106

[58] Field of Search 365/49, 106, 120, 126

[56] References Cited

U.S. PATENT DOCUMENTS

3,572,881 3/1971 Nishida 365/124

3,841,729 10/1974 Ando et al. 365/127

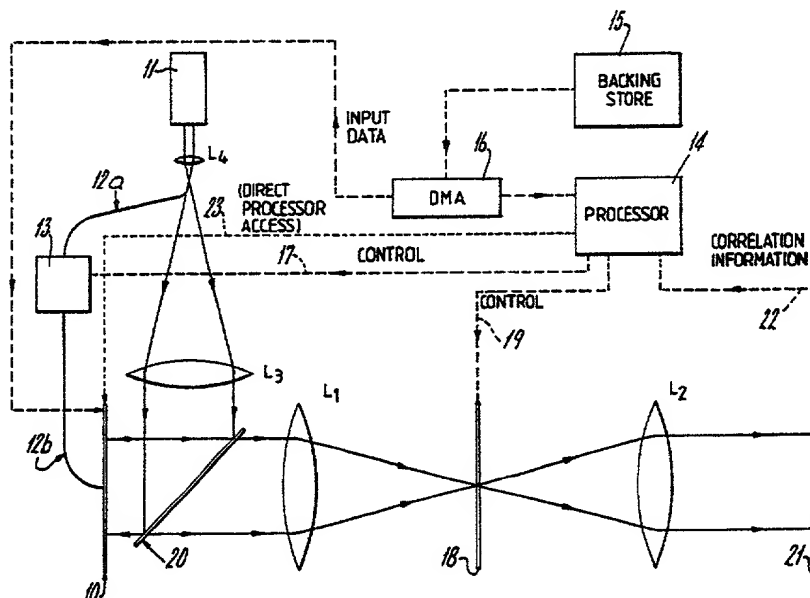
Primary Examiner—Terrell W. Fears

Attorney, Agent, or Firm—Kerkam, Stowell, Kondracki & Clarke

[57] ABSTRACT

An associative (content addressable) optical memory system is comprised by a matched optical holographic filter (10, 18, 21, L1, L2) coupled to a digital computing system (14, 15, 16) having a memory (15, 16). In order to search the computing system memory for occurrences of an item, a hologram of a binary representation of the item is formed in the Fourier transform plane (18) employing laser (11) as the light source. A page of the memory to be searched is subsequently displayed at the input plane (10), which for example is comprised by a liquid crystal over silicon display, and illuminated by the laser (11). The light is filtered at the Fourier transform plane (18). Any occurrences in the input plane display of the item result in a respective correlation spot at the output plane, which spot output is decoded and supplied to the processor (14), thus providing correlation information to the computing system as to the location in its memory of the occurrences of the item.

13 Claims, 3 Drawing Figures



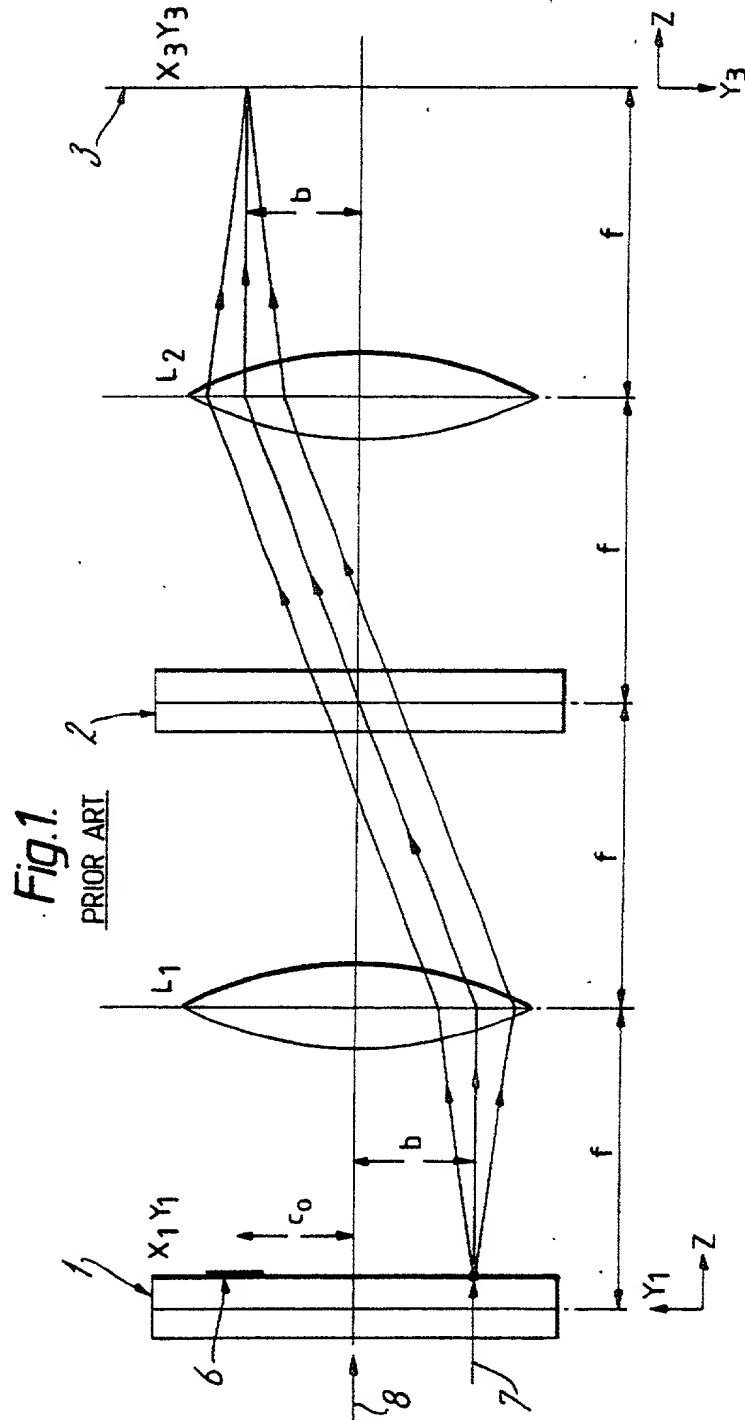
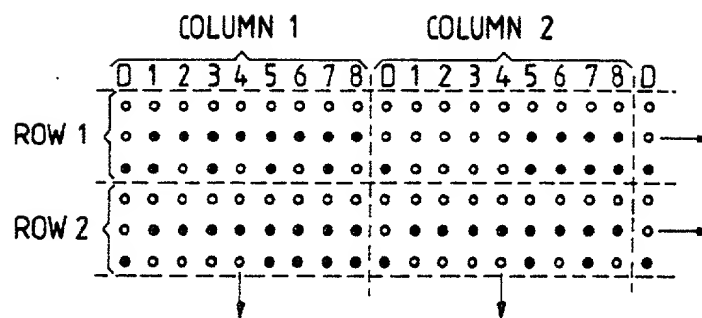


Fig. 2.

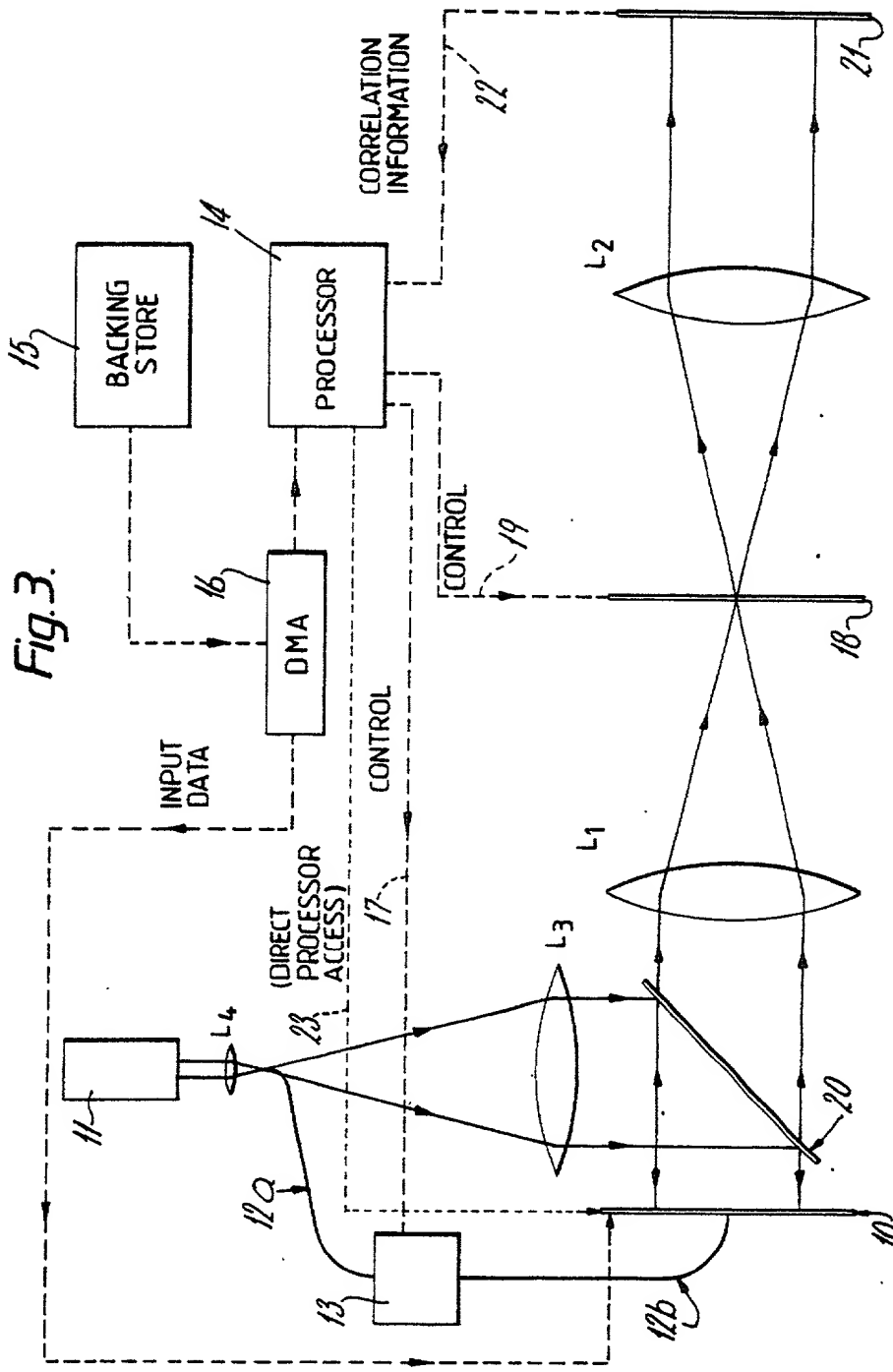


○ } = DON'T CARE
○ } (MASKED)

○ } = WORD DELIMITER
○ }

○ } = LOGICAL 0
○ }

○ } = LOGICAL ONE
○ }



ASSOCIATIVE MEMORY SYSTEMS

BACKGROUND OF THE INVENTION

This invention relates to associative memory systems, that is to say content addressable memory systems, and in particular to associative optical memory systems.

SUMMARY OF THE INVENTION

According to one aspect of the present invention there is provided an associative optical memory system comprising an optical imaging system, in the form of a matched optical holographic filter, and, coupled thereto, a digital computing system including a memory, and wherein in use for searching the computing system memory for occurrences of an item the computing system controls the input and Fourier transform planes of the filter and a coherent light source for parallel optical processing of the memory content, and the output plane of the filter provides information to the computing system as to the location in the memory of occurrences of said item.

According to another aspect of the present invention there is provided an associative optical memory system comprising a matched optical holographic filter, including an input plane comprised by a liquid crystal over silicon display, a Fourier transform plane and an output plane, the planes being separated by thin spherical lenses, and a coherent light source, and a digital computing system including a processor and a direct memory access element loaded by a backup store, which computing system is coupled to the filter for controlling the input and Fourier transform planes and the light source for parallel optical processing of the memory content, the output plane providing information to the processor as to the location of occurrences in the memory of a searched for item, wherein in use of the memory system a page of the memory content is loaded into the input plane, stored and the display blanked, a representation of the item to be recognised is loaded into a defined area of the display and a hologram thereof recorded in the Fourier transform plane by a pulsed reference beam derived from the coherent light source, wherein the loaded page is then displayed and illuminated by another beam derived from the coherent light source, wherein the light is filtered in the Fourier transform plane, detected at the output plane, decoded and passed to the processor for use in determining the location in the memory of the input display of the occurrences.

BRIEF DESCRIPTION OF THE DRAWINGS

Embodiments of the invention will now be described with reference to the accompanying drawings in which:

FIG. 1 illustrates a matched filter double Fourier transformation one-to-one imaging system (prior art);

FIG. 2 illustrates a possible representation of binary information, and

FIG. 3 illustrates a system block diagram of an associative optical memory.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Optical holographic techniques give high potential for parallel processing and while much attention has been paid to holography for mass storage of digital data,

little attention has been paid to the parallel processing of digital information by such techniques.

Data stored in an immediate access memory of a Von Neumann computer can be organised in many ways from one extreme of all data being at specified locations, to the other extreme of searching through the complete data memory to find the required item or items. Most software involving a data base of appreciable extent is organised to restrict the search by arranging the data in a logical manner, so that an algorithm can be written to reduce the search from a complete set of data items to a subset. Techniques such as arrays, indexes, buffers, linked lists, indexed sequential files are employed to achieve this aim. Nevertheless, there is a limit to what can be practically achieved by algorithms, and an element of searching item by item for the required degree of match is necessary in many applications.

With a Von Neumann architecture the memory is directly addressed by location and the searching is a step by step serial process. Associative processors, on the other hand, can generally be described as being characterised such that data can be found according to what they are, rather than the memory addresses at which they are stored, and such that logical operations can be performed over many sets of arguments at the same time with a single instruction. To do this effectively, either very high speed serial or parallel processing is required. Early associative (or content addressable) stores were high speed serial, but there is a move towards parallel processing using arrays of processors.

An example of such an array of processors is based on the transputer. Transputers are Von Neumann machines that can be connected in an array. Each machine has its own memory and searching can be done in parallel across as many machines as is desired. However, each machine searches in a serial fashion through its own memory. Clearly the efficiency of this process is dependent on the distribution of the information requiring to be searched between the processors in the array. If the search is localised to one or a few machines, the rest may be held up waiting for the answers. If the information is unduly distributed this would imply that the application is one which is suited to a wide distribution of the information. These considerations may give to conflicting design constraints affecting the array harness and the functional division between processors.

Another approach is to use what is called "data flow processors" where the order of execution of the operators of a program is determined solely by the data dependencies. It is possible for the operations to be executed in many different orders and, in particular, far more than one instruction to be executed concurrently and hence, with a large number of processors, parallelism can be exploited. The applications of this approach appear to be primarily high speed computation, rather than data base management or information handling. Reduction machines are another possibility of which there are two basic varieties, i.e. string and graph. The string reduction machines depend for their efficiency on organising the program and data in a tree structure. They are special purpose machines suited to applicative languages. The graph reduction machine is also suited to applicative language and works in terms of packets. A packet pool takes the place of a random access memory and a multiprocessor system processes those packets which contain information on the operations to be performed and the data to be used in that performance.

Most if not at all of the above approaches are tackling compute bound problems rather than memory bound problems. Signal processing and arithmetic computation are inherently compute bound (a large amount of processing on a relatively small amount of data). Data retrieval in intelligent knowledge based systems can be made into compute bound problems by complex linking of the data in, for example, a hierarchy tree structure, or they can become memory bound if simpler structures involving more searching are invoked. All problems are memory bound while the data required is on a backing store.

There is a case, therefore, for exploring solutions to memory bound problems as well as compute bound problems. While there is an element of competition between the two approaches, they should be capable of being made to compliment each other. To simplify the following, it will be assumed that the memory to be described interfaces with a conventional computer architecture, but that is not an overriding constraint.

Memory organisation at present is largely dictated by economic considerations and is hierarchical. In general, the cost of storage decreases as the access time increases. Hence the hierarchy; tape, disc (or drum), magnetic bubble, semiconductor, in increasing cost per bit and decreasing access time. All of these memories are serial access in terms of words, bytes or bits. There is no memory that is parallel in terms of more than a few multiple words.

The normal method of retrieving information that is stored outside of the immediate access memory, is for the central processor to request a peripheral to put into the immediate access memory a specified file, record or set of records. The central processor keeps some sort of index of the whereabouts of the relevant information and the peripheral usually searches and finds the exact information and loads it. This can be done autonomously if the processor has a DMA (direct memory access) feature.

Once loaded, a more detailed search is often necessary to find a file with a particular value in a set of data items. Such a search might advantageously be carried out as a parallel operation over a number of fields simultaneously.

Optical techniques are potentially capable of a very high level of parallel processing and have been used for some years in the processing and recognition of visual analogue information. Optical techniques are also used for the storage of digital information. However, optical techniques appear not to have been used for the recognition of digital information.

Optical storage of digital information may be divided into two broad categories: (a) using holographic techniques and (b) using direct recording of binary information. Much work has been done on the former with very little commercial success. The latter, because of its commonality with home video recording, shows considerable prospect of providing storage media which is a competitor to magnetic disc. Direct recording lends itself to serial read out or limited parallel read out of information, whereas holography lends itself to full parallel read out. Direct recording is a surface recording. Holographic recording can be either a surface or a volume recording, and in the latter case can lead to very high information densities. Direct recording is very sensitive to local imperfections in the vicinity of the recording medium (e.g. dust) whereas holography is not. Both approaches use recording media that are more

suited to read only rather than read/write storage. Although both techniques involve lasers, holography is much more dependent on a coherent light source. Holographic equipment is bulky because it usually involves lenses with long focal lengths and needs very precise and stable dimensioning. Direct recording needs even more precise and stable dimensioning and usually involves rotating parts but the optical system is simpler. The reason that direct recording has been used for video recorders in the home entertainment market is probably because of the simpler optics and the lack of suitable detectors to cope with highly parallel outputs. These considerations do not weigh so heavily for data processing.

The present invention is not, however, concerned with high density optical storage, for which there are many possibilities not necessarily using holography, but rather with the parallel processing of information using holographic techniques, and carrying parallel processing one stage further than hitherto from the memory towards the processor so that parallelism can be used to provide rapid search facilities for matching fields in relatively complex data structures or for individual items.

The basis of much work on optical pattern recognition is the matched filter, sometimes called a "Vander Lugt" filter, illustrated in FIG. 1. The filter contains three planes 1, 2 and 3 all of which are situated at the foci of two thin spherical lenses L1 and L2. Plane 1 is called the input plane, plane 2 is called the spatial frequency plane or Fourier transform plane (FTP) and plane 3 is called the output plane. Lense L1 acts to provide the Fourier transform of images in the input plane 1 at the Fourier transform plane 2, and the lense L2 acts to provide the inverse Fourier transform of images at the Fourier transform plane 2 and the output plane 3. The Fourier transform of a real image at the input plane 1 which is illustrated by coherent light is an interference pattern or hologram. The input plane 1 contains spatial information in two dimensions, and the FTP, plane 2, contains spatial frequency information in two dimensions. Just as an electrical signal can be broken into frequency components by a Fourier transform, visual patterns can be treated likewise. Electrical signals are, however, usually one dimensional, whereas optical information is two or three dimensional. The concern here is with two dimensional optical information, such as displayed on a page of print, for example.

The first stage in the production of a filter is to record a hologram in the FTP, plane 2, corresponding to the information to be recognised. To do this a representation of the information to be recognised (image 6) is placed in the input plane 1, centred, say, at c_0 and a point source reference beam indicated by arrow 7 is located at $x=0$, $y_1=-b$. The coherent light source used to illuminate the image 6 is coherent with the reference beam 7 and hence differs from it only in amplitude and phase. The resultant hologram in the FTP is recorded, for example, on a photo sensitive medium as a plane hologram transparency. The lens L2 and the output plane 3 are not required for this process.

Having recorded the hologram of the information to be recognised, it being disposed in the FTP, plane 2, a set of unknown patterns is placed in the input plane 1 and illuminated by coherent light of the same wavelength as is used in the recording process. Now, a basic property of a hologram is that if it is recorded by illumination of two scenes S1 and S2 and then exposed to

light from S1, a real or virtual image of S2 will result, and vice versa. In the matched filter a real image is produced at the output plane 3. S1 can be regarded as the point source, and S2 initially as the information being recorded and latterly as the unknown pattern. If the unknown pattern contains the information to be recognised, the result will be a representation of the point source in the output plane 3. This represents the autocorrelation, or matching, of the input to the filter. With suitable design, the autocorrelation spot will be more intense than any cross correlation terms and a peak detector can determine its presence and position in the output plane. Its position in the output plane bears a one-to-one correspondence with its position in the input plane. For example, if its position in the input were to be exactly the same as the position when the hologram was made, the spot would be at $x=0$, $y_3=-b$ (note the inversion of the axes). This can be regarded as the reference point and the displacement of the spot from that reference point is a direct measure of its displacement in the input scene from the original position.

There are a number of strategies for making use of matched filter techniques for optical character recognition, as discussed for example in "Optical Holography" R. J. Collier, C. B. Burckhardt, L. H. Lin, Academic Press 1971. The strategy depends the need to recognise:

- (a) The position of items x_1, x_2, \dots, x_n in a scene made up from a set of n items with an arbitrary number of repetitions of each item (e.g. printed character recognition).
- (b) The presence and position of one item against an unspecified background (e.g. military applications of tank detection).
- (c) The presence of a number of examples of one item from a set of n items (e.g. searching for one binary pattern in a set of n binary patterns).

Although (a) is more general than (c), there are practical reasons why (even when the application requires it) searching for items one at a time rather than many together may be preferable. However, (c) implies a different matched filter for each item sought and it is then necessary to be able to change matched filters very rapidly if this approach is to be adopted effectively.

A basic problem of this type of optical processing is, therefore, that while the processing is very fast once the information is in the optical filter, actually getting it into position may provide an unacceptable bottleneck.

The most widely used holographic recording material is silver halide photographic emulsion. This is satisfactory from the point of view of resolution and contrast range but needs chemical processing and implies complex mechanical handling problems if filters are to be changed.

There are many other possibilities, some of which do not require chemical processing, for example ferro electric crystals, inorganic photochromic materials, thermoplastic materials (require a developing process which is, however, fast by photographic standards), magneto-optic materials, ferroelectric photoconductor, liquid crystal/photoconductor and thermally-written smectic liquid crystal. Our co-pending GB Application No. 840482 (Ser. No. 699,980 filed 2/8/85) (W. A. Crossland—R. W. A. Scarr 44-31) relates to optical processors using a transmission type smectic liquid crystal display in the FTP.

Recording holograms by optical means is not, however, the only possibility. It is also possible to generate holograms by a computer if the patterns to be handled

are of sufficient simplicity. Binary patterns can be regarded as arrays of point sources. Such holograms may be generated once, stored in binary form and loaded as required onto a suitable display. Alternatively they could be computed at run time and loaded. However, the time necessary to load the information, if this is done serially may negate advantages obtained by subsequent parallel processing.

For the purposes of the following description the holograms will be assumed to be recorded by optical means in an undefined recording media. This being so, together with the fact that a very fast means of production is required, implies that (a) the hologram is recorded by a transient laser pulse of high energy and short duration (microsecond or submicrosecond), (b) that hologram is read by a transient pulse which may be of lower energy than the recording pulse, (c) the read is preferably but not necessarily non-destructive, (d) the hologram storage need not be permanent and can decay in times of the order of milliseconds, (e) the duty cycle of the laser, particularly on recording, is sufficiently low that means power level limitations are not exceeded, and (f) erasure and recording may be one operation or two depending on the recording medium.

The parallel processing of a "page" of memory displayed at the input plane will now be considered, the memory "page" being equivalent to the set of unknown patterns referred to above. The entries on the "page" are in the form of "words" of n bits. For the purposes of illustration n will be taken as 8, but in principle it can be any number. Ideally any concatenation of words should be recognisable and it should be possible to mask bits, for example one may be interested in the last m bits of a word and the first $n-m$ bits are of no account and can be "masked out". There are thus four conditions to be recognised: a logical 0 a logical 1, a don't care and delimiters or a frame defining the boundaries of a word. It is necessary to have an arrangement that will cope with all bit patterns without aliasing, that is there must be a zero possibility of a pattern being generated and registered correctly that is an unintentional representation of the one being sought, thus implying a fairly high order of redundancy. The problem that a different pattern in the y dimension might alias the required pattern in the x direction can be avoided by recording and processing in the presence of a fixed but limited pattern, possibly around the edge of the display.

FIG. 2 shows an example of a word design in which a computer word of 8 bits is represented by 27 bits on the display, the invention, however is not to be considered as limited to 8 bit words, any length may be used. A black dot represents the presence of illumination and a clear dot its absence. The 8 bit words are arranged in rows and columns, only four words being indicated in FIG. 2. Each word is comprised by three sub-rows and nine sub-columns and begins with a column word delimiter (sub-column D) comprised by a sub-column of two clear dots followed by a black dot as indicated. Each first sub-row of a word is comprised by nine clear dots, thus acting as a row word delimiter. To facilitate understanding of FIG. 2 dashed lines have been drawn to delimit the four words thereof. The word occupying the row 1, column 1 is (10101010) i.e. hexadecimal AA; that in row 1, column 2 is (MMMM1111) where M means masked i.e. hexadecimal XF where X is any value; that in row 2, column 1 is (00001111) i.e. 0F and that in row 2, column 2 is (00001010) i.e. 0A.

The "page" of memory in the input plane can be loaded from a computer's electrical memory, in which case the computer will keep a map relating the contents of the input plane to its internal memory. Such a map would be a simple linear translation from one medium to the other. Alternatively, the page of memory may be the direct output of an optical (holographic) store projected onto the input plane. If so, the information would be processed as described below, and then loaded into an electrical memory if an item being sought was present in the page being displayed. A third possibility is the direct loading of the input plane from a magnetic backing store using direct memory access (DMA). Here there is the possibility of either loading an immediate access memory afterwards if an item is found, or loading the immediate access memory is parallel with loading the input plane. Input displays that take the form of liquid crystal over silicon can store the information for display and allow simultaneous read access to it by a processor.

The interface between the backing store and the display (input plane) must allow for the fact that the binary information needs to be displayed in a redundant format and must provide the necessary translation arrangement. Direct interfacing to a holographic store implies that the information in that store would be in the same redundant format as is required for the display. By selecting the appropriate fields (e.g. the bottom elements of each row and the relevant columns in FIG. 2), the information can be read out of a liquid crystal display store as pure binary information.

The matched filter, which is to be put in the FTP, is a recording of the word pattern sought together with its delimiters. If masking of words is required this is done on recording. The presentation of the word to be recorded can be placed at any arbitrary point on the display (input plane) although on the basis of symmetry it is sensible to place the reference beam at the exact centre of the display and on the focal axis of the system ($x=y=0$). The intensity of the reference beam should be much greater than that of the other individual point sources representing the binary numbers, in order that the autocorrelation terms be larger than the cross correlation terms at the output plane.

The output plane may contain an array of photodetectors or be comprised by a homogeneous photosensitive plane such as provided for a TV camera. If photodetectors are used, there is one for every word stored on the input plane located at a position corresponding to the possible position of the words in the input plane. It is assumed that word boundaries are fixed points at the input plane and information is, therefore, fixed within a grid set by these word boundaries. If a photosensitive plate is used its photosensitive storage media would be scanned to locate the amplitude and position of maxima in the output plane. In either case a replica of the word being sought may be placed at a known point on the input plane so as to act as a reference for determining the level at which to set the correlation threshold. This enables correction to be made automatically for variation in the intensity of illumination of the input plane.

The positions of valid maxima in the output plane are conveyed to the processor which translates them into addresses in its own memory, which may also be the input plane, where the items identified can be found and processed.

The only logical operation the arrangement is capable of performing is "search on masked equality".

Whereas semiconductor logic in memory arrangements are capable of such operations as searching for less than, or greater than, and performing logical and arithmetic operations on operands when found, an optical arrangement can only locate the relevant locations, however, it can do these in combinations which the semiconductor logic in memory arrangements is less well able to do. For example, a set of personnel records might be scanned as one operation to find all the persons named Brown with two children and red hair.

From consideration of the search times of a conventional Von Neumann machine and of an optical correlator attached to a Von Neumann machine it is apparent that a very fast means of producing holograms, of the microsecond order, is required; that the optical correlator is most disadvantaged when the information to be scanned is already in an immediate access memory and a single scan of the material is required; that the optical correlator is advantaged when the information is on a backing store and multiple passes or combinations of words in fixed relationships are sought; and that the optical correlator is further advantaged when the backing store material is already in an optical format. In drawing these conclusions it was assumed that, when the information is on a backing store, the limitation is set by the speed of reading that store rather than any limitation on loading the information into the input plane. It is also assumed that processing of information at the output plane can be done at speeds comparable with a conventional processors memory cycle time.

Compared with the resolution required for holographic optical lenses, the resolution requirements of the recording in the FTP are not onerous. Furthermore, there is no basic requirement to work in the visible range and operation in the infra red can be beneficial if resolution of the holographic recording media is limited.

There are a wide range of possible laser sources which may be used for producing holograms in the FTP. The parameters of a laser which need to be considered are the peak power under pulsed operation and duty cycle, wavelength and spectral distribution of energy. In general, a very highly coherent source is not compatible with high peak power outputs, and it is considered that the high coherence required for monomode communications systems is not necessary. Consideration of hologram resolution points to a source in the infra red rather than the ultra violet. Nd:YAG lasers provide outputs in the $1\text{ }\mu\text{m}$ region and CO_2 lasers in the far infra red, about $10\text{ }\mu\text{m}$.

Photodetection in an array or a homogenous photosensitive plate have been referred to above for use in the output plane for detecting correlation peaks. An array of diodes is presently considered preferable since they are all solid state and the processing of their outputs is relatively straightforward. The diode array may be fabricated on a single chip. The density of the detectors would be relatively low, only one being needed per word, so that chip utilisation would be poor. However, the chip could also contain processing circuitry to present correlation peak positions across a defined interface. All of the diodes on the chip would need to have uniform sensitivity and whilst silicon would be ideal from a fabrication point of view the fall-off in its photosensitivity at the infra red end of the spectrum could be disadvantageous.

The above description has covered various approaches to the design of an associative optical mem-

ory, that is to say an associative memory arrangement of large capacity based on the use of a method holographic filter connected to a digital computing system, a specific system incorporating presently preferred options will now be described with reference to the block diagram of FIG. 3.

An input plane is comprised by a liquid crystal over silicon display 10 incorporating a pleochroic dye and working in a similar manner to that described in British Patent Application No. 8209710 (Ser. No. 2118247A) (W. A. Crossland et al 35-13-7-5). A major difference, however, is that the reference light source derived from the output of laser 11 is coupled by an optical fibre 12a, 12b through a hole in the centre of display 10. A means of turning the reference light source on or off is required and is illustrated as an optical modulator 13 in series between fibre sections 12a and 12b. The optical modulator may be a solid state electro-optic device. Alternatively the reference light source may be switched on or off by some form of liquid crystal device employed as a shutter. The system includes lens L1 and L2 equivalent to lenses 4 and 5, respectively, of FIG. 1.

To record a hologram a processor 14 loads a representation of the bit pattern to be recognised into a defined area of the input display 10. The rest of the input display is blanked, its content (the input page of memory to be searched) having been previously loaded, from backing store 15 via DMA 16, and stored. The optical modulator 13 is switched on via control line 17. If necessary the processor 14 sets the material in the FTP 18 and comprising a dynamic hologram recording medium into the "ready for record" state, as for example in the case of a ferroelectric crystal, via control line 19. The laser 11 under command by the processor 14 gives a high energy pulse (reference beam) to illuminate the representation of the bit pattern to be recognised (via lens L4, fibre 12a, 12b and modulator 13) and the hologram thereof is recorded in the FTP 18. The optical modulator 13 is then switched off by command of the processor whereby to turn the reference beam off at the display 10.

On command from the processor the input page of memory to be searched is displayed on display 10 and the laser 11 gives a low energy output pulse which via lens L4, lens L3 and beam splitter 20 illuminates the input page on display 10. Lenses L3 and L4 comprise a beam collimator. The light is filtered in the FTP 18 and detected by photosensitive matrix, e.g. a silicon diode array, at the output plane 21. The output information is decoded and passed back to the processor as correlation information 22, which processor 14 can then locate in the memory of the input display, the DMA 16 being coupled to the processor as shown, the items that have been recognised.

When the processor has finished processing the information in the input plane, which may involve several cycles as outlined above, each cycle being employed for locating a respective item, the input plane (display 10) is reloaded with fresh material via the DMA 16 from the backing store 15. The processor is free to carry out other operations while the DMA Loading is taking place.

The data (memory) page capacity of the system can be limited by optical resolution, or fabrication limitations, in any one of the three planes i.e. the input, the Fourier transform and the output plane. However, because the output plane resolution that is required is one bit per word, this is unlikely to be the limitation. The

size of the display (input plane) is limited if, as proposed, a liquid crystal over monocrystalline silicon display is used. A four inch (10 cm) wafer of such material might allow some 50 square cm of useful area. If, as in British Application No. 8208710, a transistor and a grid of connections are required behind each element in two out of three rows, assuming the arrangement of FIG. 2, the minimum size of each element will be, say, 10 micron square. This would give a potential for 5×10^7 elements, corresponding to 1.4×10^7 bits of stored information (8 bit words).

The theoretical limit for storage in the FTP is of the order of 18×10^7 elements, or 5.3×10^7 bits of stored information, assuming light of a wavelength of 1 micron, no limitation set by hologram definition, the same 50 square cm area for storage, and a plane hologram.

The practical limit for data page capacity is therefore determined by the input plane, where an ability to fabricate successively a number of elements of the 10^7 order is beyond the current state of the art. Techniques involving redundancy and selection on test as used on bubble memories may ease this problem, but are limited by the fact that information being searched for must always have the same relative geometry irrespective of its location on the display.

If capacity is limited at the input plane rather than the FTP, the area of the FTP can be less than that of the input plane, implying an asymmetrical lens L1 in FIG. 3, thus reducing the power requirements needed to produce the hologram.

The processor has the ability to access directly a liquid crystal over silicon version of the input plane, which acts as its memory, as mentioned above, this alternative being indicated by the dotted line "direct processor access" 23 between processor 14 and plane 10.

Thus the present invention proposes an associative optical memory system and the requirements of the component parts of such an arrangement have been outlined. Such systems lend themselves to the parallel processing of millions of bits in the search for a particular item of information. The associative memory concept as outlined is largely complementary to advanced processor architecture and is relevant to 5th generation computers with extensive knowledge/data bases.

I claim:

1. An associative optical memory system comprising an optical imaging system, a coherent light source and a digital computing system including a memory, the optical imaging system being in the form of a matched optical holographic filter having an input plane, a Fourier transform plane and an output plane, the computing system being coupled to the filter and including means for controlling the input plane, the Fourier transform plane and the coherent light source for parallel optical processing of the memory content in order to search for occurrences of an item in the memory, the output plane having an output providing information to the computing system controlling means as to the location in the memory of occurrences of said item, said controlling means serving to form a hologram of said item in the Fourier transform plane, using said coherent light source as the light source for the hologram formation, and the controlling means serving subsequently to display the memory content at the input plane and to illuminate it with said coherent light source, the Fourier transform plane serving to filter light from the input plane, any occurrence in the input plane display of said

item resulting in a corresponding output in the output plane which is detected and provided as said information to the computing system means.

2. A memory system as claimed in claim 1, wherein a processor of the computing system comprises said computing system controlling means, and wherein said memory includes a direct memory access element loaded by a backing store.

3. A memory system as claimed in claim 1, wherein the coherent light source is comprised by a laser the optical output of which can be supplied to illuminate the input plane or which can be supplied as a reference beam to the input plane.

4. A memory system as claimed in claim 3, wherein the reference beam is supplied to the input plane via an optical fibre including an optical modulator therein.

5. A memory system as claimed in claim 1 wherein the input plane is comprised by a liquid crystal over silicon display.

6. A memory system as claimed in claim 1, wherein the input plane is comprised by a liquid crystal over silicon display which also comprises the digital computing system memory.

7. A memory system as claimed in claim 1, wherein the Fourier transform plane is comprised by ferroelectric crystal or thermally written smectic liquid crystal.

8. A memory system as claimed in claim 1, wherein the output plane is comprised by an array of photodiodes.

9. A memory system as claimed in claim 1, wherein said item is comprised as a specific binary pattern or patterns, wherein the memory is configured in the input plane as a plurality of binary patterns, and wherein said matched filter serves to locate the specific binary pattern or patterns against a background of the plurality of binary patterns.

10. A memory system as claimed in claim 9, wherein the binary patterns are defined with a redundant coding scheme that permits discrimination without aliasing.

11. A memory system as claimed in claim 9, wherein each binary pattern comprises a computer word of 8 bits represented at the input plane as 27 bits comprised by three rows of nine bits, the first row and first column of which serve to delimit words from one another.

12. A memory system as claimed in claim 11 and wherein the columns other than the first represent logical 0, logical one or don't care (masked).

13. An associative optical memory system comprising a matched optical holographic filter, including an input plane comprised by a liquid crystal over silicon display, a Fourier transform plane and an output plane, the planes being separated by thin spherical lenses, and a coherent light source, and a digital computing system including a processor and a direct memory access element loaded by a backup store, which computing system is coupled to the filter for controlling the input and Fourier transform planes and the light source for parallel optical processing of the memory content, the output plane providing information to the processor as to the location of occurrences in the memory of a searched for item, wherein in use of the memory system a page of the memory content is loaded into the input plane, stored and the display blanked, a representation of the item to be recognised is loaded into a defined area of the display and a hologram thereof recorded in the Fourier transform plane by a pulsed reference beam derived from the coherent light source, wherein the loaded page is then displayed and illuminated by another beam derived from the coherent light source, wherein the light is filtered in the Fourier transform plane, detected at the output plane, decoded and passed to the processor for use in determining the location in the memory of the input display of the occurrences.

* * * * *

40

45

50

55

60

65

United States Patent [19]

[11] Patent Number: 5,479,488

Lennig et al.

[45] Date of Patent: Dec. 26, 1995

[54] METHOD AND APPARATUS FOR
AUTOMATION OF DIRECTORY
ASSISTANCE USING SPEECH
RECOGNITION[75] Inventors: **Matthew Lennig; Robert D. Sharp,**
both of Westmount; **Gregory J. Bielby,**
Pointe-Claire, all of Canada[73] Assignee: **Bell Canada,** Montreal, Canada[21] Appl. No.: **193,522**[22] Filed: **Feb. 8, 1994**

[30] Foreign Application Priority Data

Mar. 15, 1993 [CA] Canada 2091658

[51] Int. Cl.⁶ **H04M 3/64; G10L 9/06**[52] U.S. Cl. **379/67; 379/213; 395/2.61;**
395/2.84[58] Field of Search 395/2.49, 2.56,
395/2.61, 2.84; 379/142, 67, 88, 89, 213

[56] References Cited

U.S. PATENT DOCUMENTS

4,696,039	9/1987	Doddington	381/46
4,790,003	12/1988	Kepley et al.	379/88
4,797,910	1/1989	Daudelin	379/67
4,831,550	5/1989	Katz	395/2
4,881,261	11/1989	Oliphant et al.	379/215
4,896,345	1/1990	Thorne	379/67
4,979,206	12/1990	Padden et al.	379/67
5,014,303	5/1991	Velius	379/201
5,018,201	5/1991	Sugawara	395/2
5,163,081	11/1992	Wycherley et al.	379/52
5,181,237	1/1993	Dowden et al.	379/88
5,267,304	11/1993	Slusky	379/213
5,335,266	8/1994	Richardson, Jr. et al.	379/112
5,339,352	8/1994	Armstrong et al.	379/112
5,357,596	10/1994	Takebayashi et al.	395/2.84

FOREIGN PATENT DOCUMENTS

1162336 2/1984 Canada 364/412

OTHER PUBLICATIONS

A*-*Admissible Heuristics for Rapid Lexical Access*, P. Kenny, R. Hollan, V. Gupta, M. Lennig, P. Mermelstein and D. O'Shaughnessy INRS-Télécommunications, Montreal, Quebec, Canada. 1991 IEEE.*Multiple-Level Evaluation of Speech Recognition Systems*, John F. Pitelli, David Lubensky, Benjamin Chigier, and Hong C. Leung; Speech Technology Group Artificial Intelligence Laboratory, Tu.SPM. 1.3, pp. 165-168.*Characterization of Directory Assistance Operator-Customer Dialogues in AGT Limited S. M. (Raj) Ulagaraj AGT Limited, Tu.SPM. 3.2, pp. 193-196.**Rejection and Keyword Spotting Algorithms for a Directory Assistance City Name Recognition Application*, Benjamin Chigier, Speech Technology Group 1992 IEEE, pp. II-93 to II-96.*Flexible Vocabulary Recognition of Speech*, Matthew Lennig, Douglas Sharp, Patrick Kenny, Vishwa Gupta and Kristin Precoda, Bell Northern Research and INRS-Télécommunications; Tu. fPM. 1.3, pp. 93-96.

Lubensky, "Word Recognition using Neural Nets, Multi-State Grossian and K-Nearest Neighbor Classifiers", 1991 IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 1 pp. 141-144.

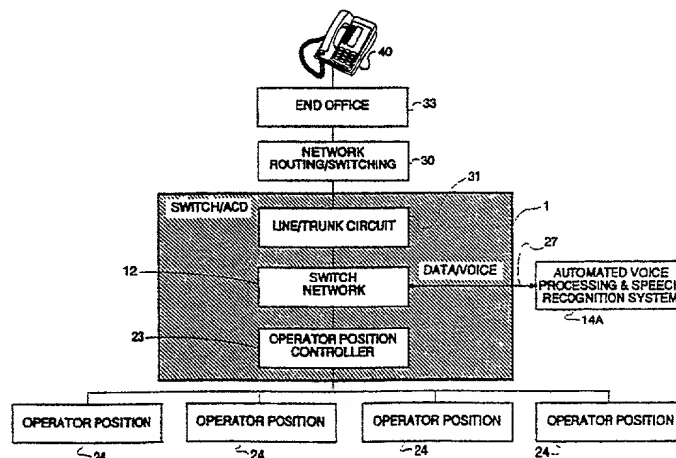
Primary Examiner—Jeffery A. Hofsass

Assistant Examiner—Daniel S. Hunter

Attorney, Agent, or Firm—Thomas Adams

[57] ABSTRACT

In a telecommunications system, automatic directory assistance uses a voice processing unit comprising a lexicon of lexemes and data representing a predetermined relationship between each lexeme and calling numbers in a locality served by the automated directory assistance apparatus. The voice processing unit issues messages to a caller making a directory assistance call to prompt the caller to utter a required one of said lexemes. The unit detects the calling number originating a directory assistance call and, responsive to the calling number and the relationship data computes a probability index representing the likelihood of a lexeme being the subject of the directory assistance call. The unit employs a speech recognizer to recognize, on the basis of the acoustics of the caller's utterance and the probability index, a lexeme corresponding to that uttered by the caller.



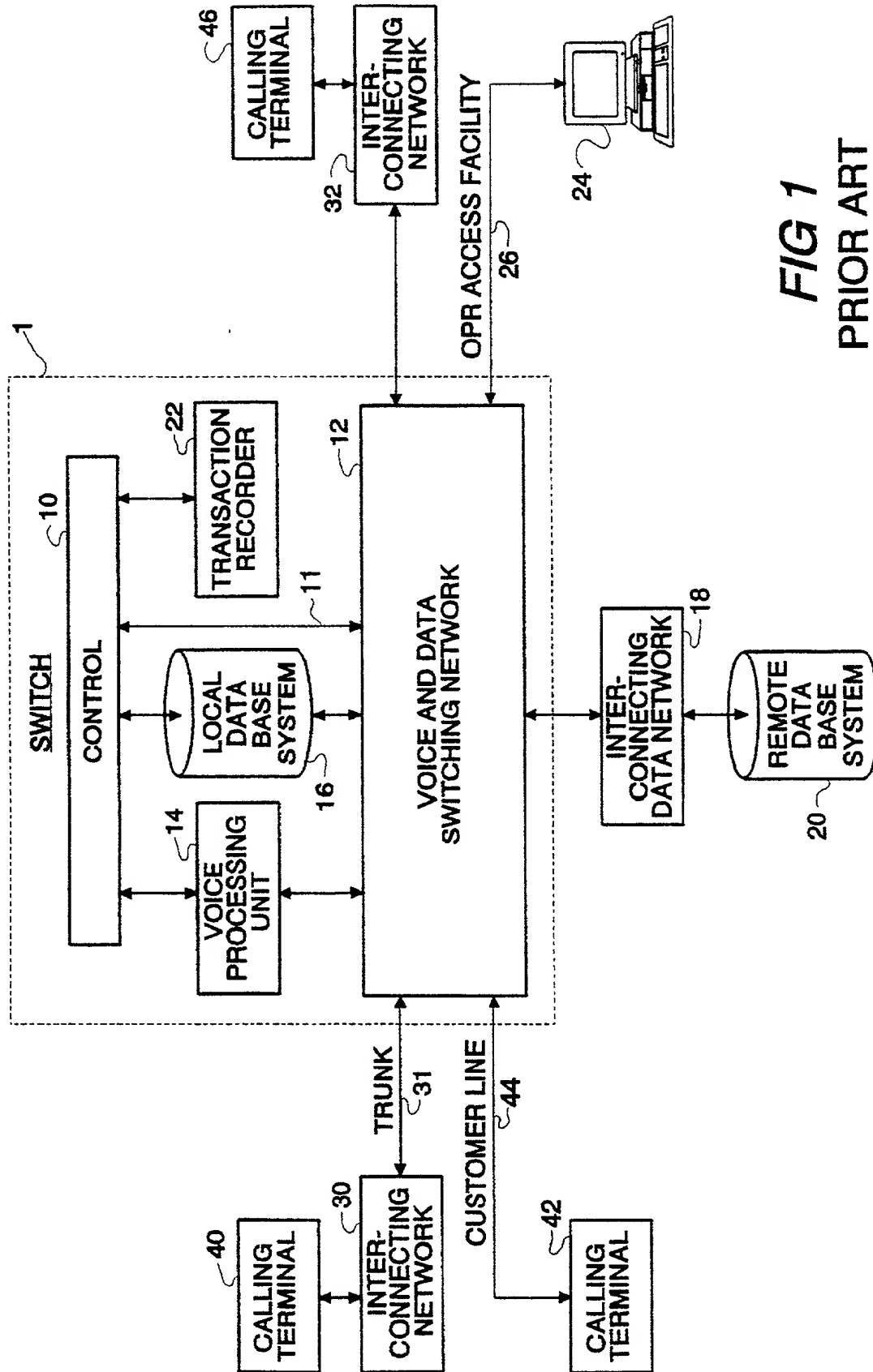
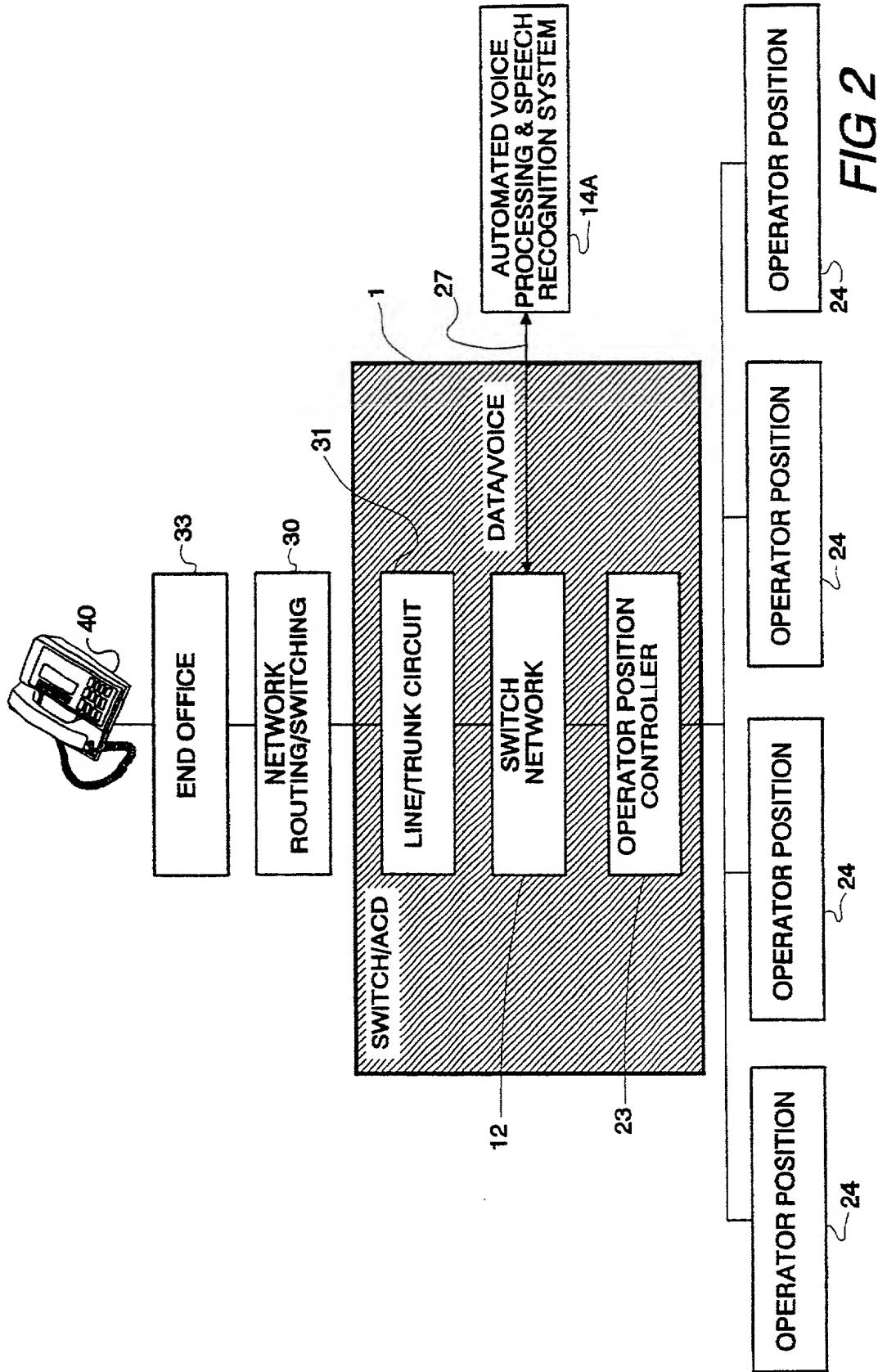
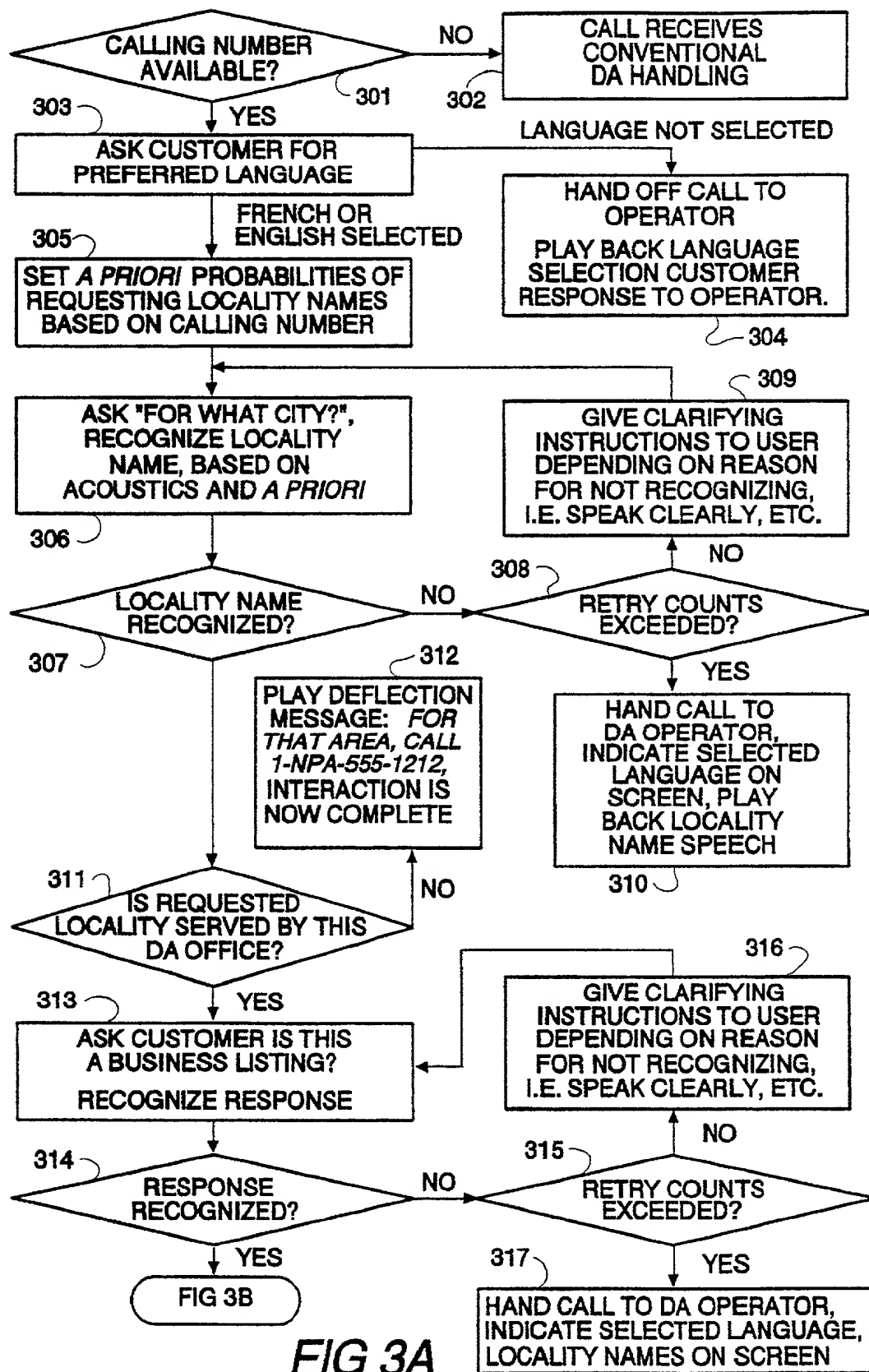


FIG 1
PRIOR ART





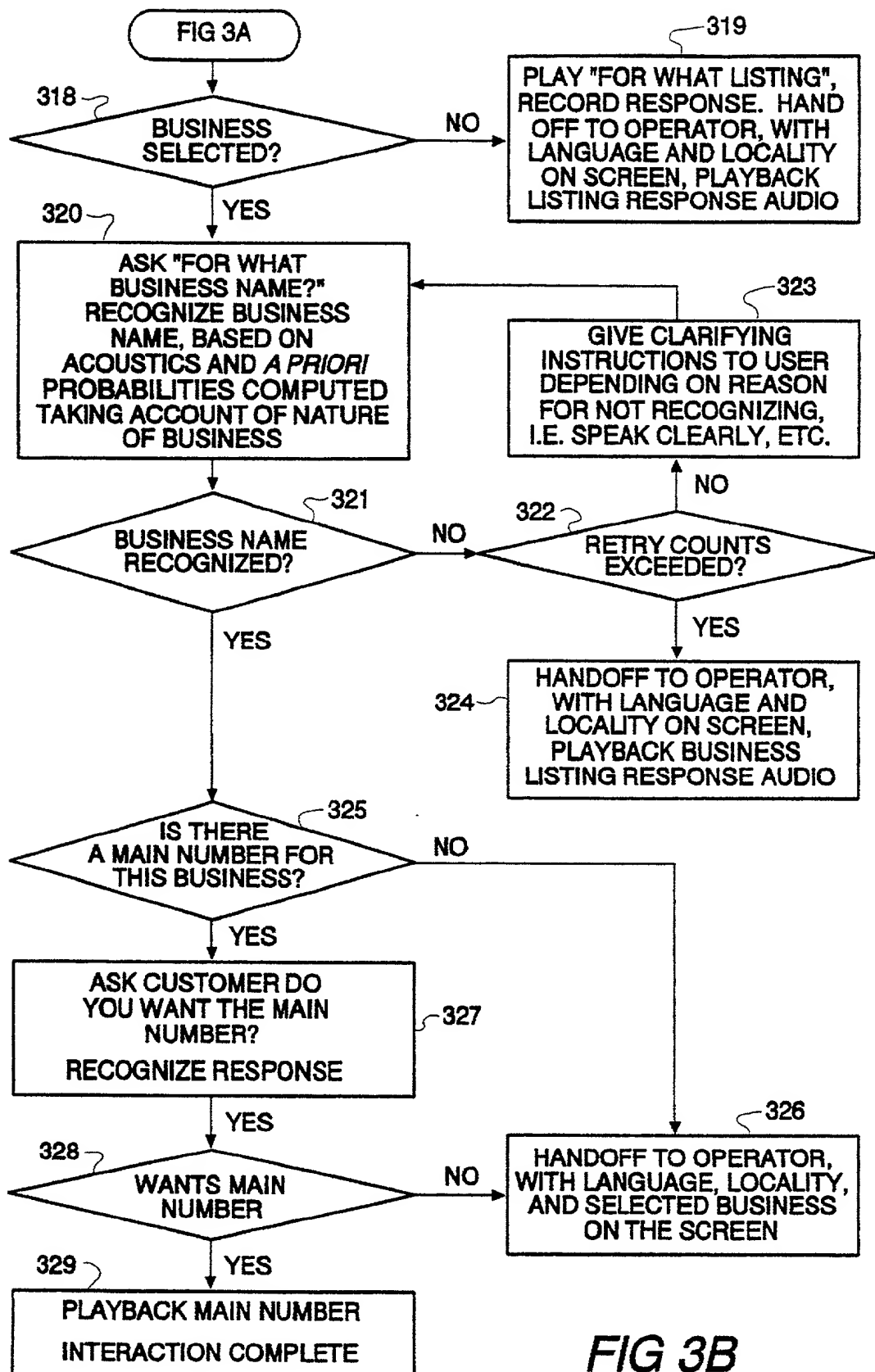
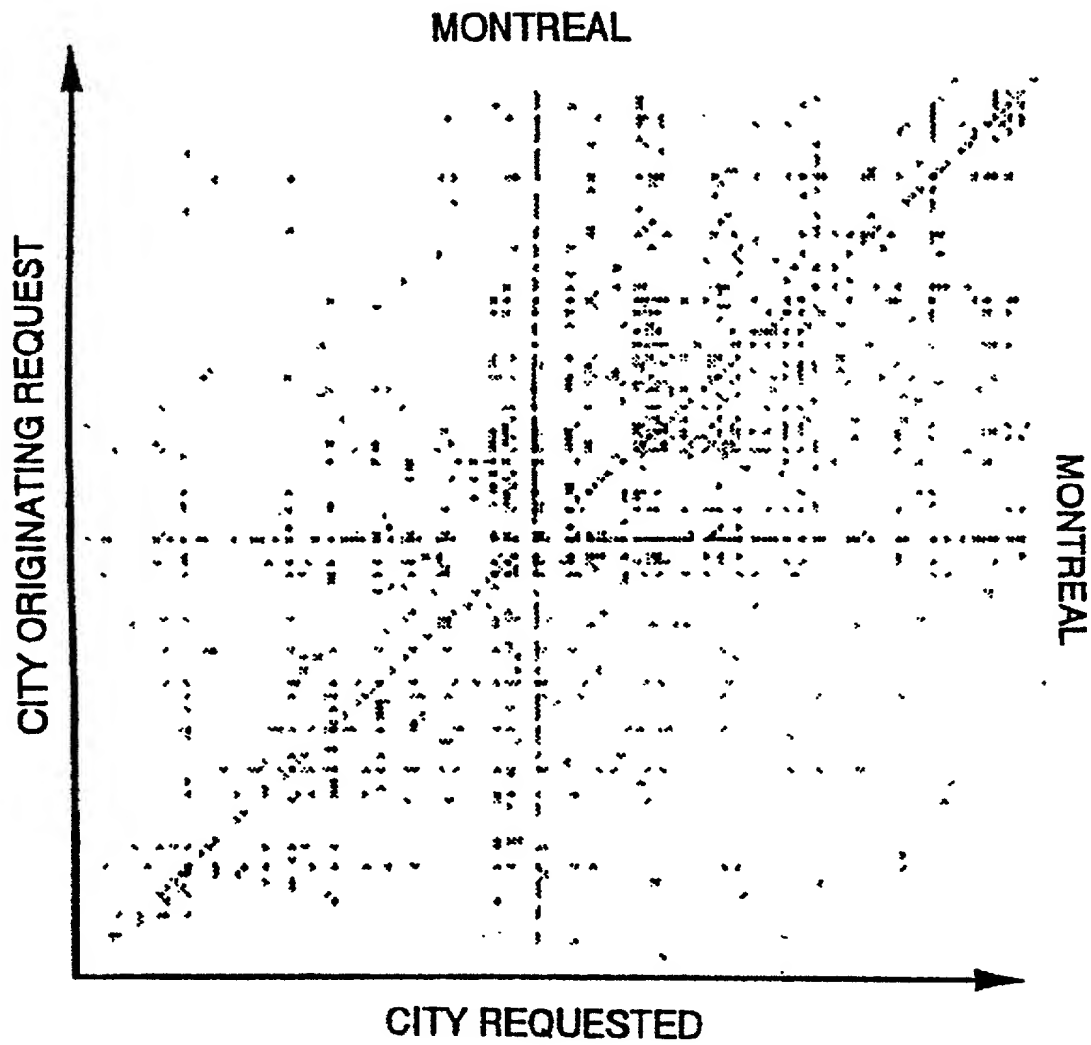
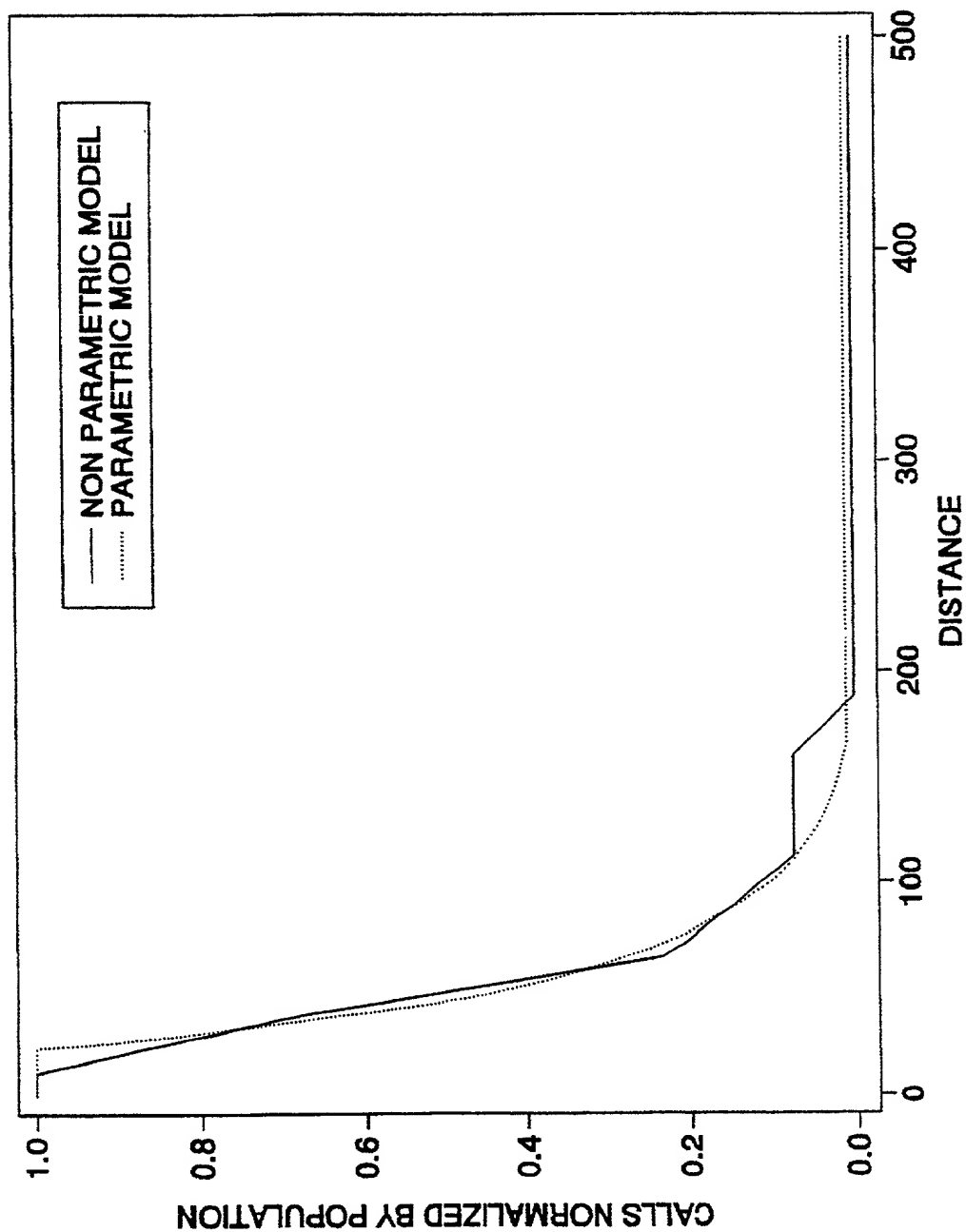


FIG 3B

**FIG 4**



METHOD AND APPARATUS FOR AUTOMATION OF DIRECTORY ASSISTANCE USING SPEECH RECOGNITION

BACKGROUND OF THE INVENTION

1. Technical Field

The invention relates to a method and apparatus for providing directory assistance, at least partially automatically, to telephone subscribers.

2. Background Art

In known telephone systems, a telephone subscriber requiring directory assistance will dial a predetermined telephone number. In North America, the number will typically be 411 or 555 1212. When a customer makes such a directory assistance call, the switch routes the call to the first available Directory Assistance (DA) operator. When the call arrives at the operator's position, an initial search screen at the operator's terminal will be updated with information supplied by the switch, Directory Assistance Software (DAS), and the Operator Position Controller (TPC). The switch supplies the calling number, the DAS supplies the default locality and zone, and the TPC supplies the default language indicator. While the initial search screen is being updated, the switch will connect the subscriber to the operator.

When the operator hears the "customer-connected" tone, the operator will proceed to complete the call. The operator will prompt for locality and listing name before searching the database. When a unique listing name is found, the operator will release the customer to the Audio Response Unit (ARU), which will play the number to the subscriber.

Telephone companies handle billions of directory assistance calls per year, so it is desirable to reduce labour costs by minimizing the time for which a directory assistance operator is involved. As described in U.S. Pat. No. 5,014,303 (Velius) issued May 7, 1991, the entire disclosure of which is incorporated herein by reference, a reduction can be achieved by directing each directory assistance call initially to one of a plurality of speech processing systems which would elicit the initial directory assistance request from the subscriber. The speech processing system would compress the subscriber's spoken request and store it until an operator position became available, whereupon the speech processing system would replay the request to the operator. The compression would allow the request to be replayed to the operator in less time than the subscriber took to utter it. Velius mentions that automatic speech recognition also could be employed to reduce the operator work time. In a paper entitled "Multiple-Level Evaluation of Speech Recognition Systems", the entire disclosure of which is incorporated herein by reference, John F. Pitrelli et al disclose a partially automated directory assistance system in which speech recognition is used to extract a target word, for example a locality name, from a longer utterance. The system strips off everything around the target word so that only the target word is played back to the operator. The operator initiates further action.

U.S. Pat. No. 4,797,910 (Daudelin) issued Jan. 10, 1989, the entire disclosure of which is incorporated herein by reference, discloses a method and apparatus in which operator involvement is reduced by means of a speech recognition system which recognizes spoken commands to determine the class of call and hence the operator to which it should be directed. The savings to be achieved by use of Daudelin's

speech recognition system are relatively limited, however, since it is not capable of recognizing anything more than a few commands, such as "collect", "calling card", "operator", and so on.

These known systems can reduce the time spent by a directory assistance operator in dealing with a directory assistance call, but only to a very limited extent.

SUMMARY OF THE INVENTION

The present invention seeks to eliminate, or at least mitigate, the disadvantages of the prior art and has for its object to provide an improved new directory assistance apparatus and method capable of reducing, or even eliminating, operator involvement in directory assistance calls.

To this end, according to one aspect of the present invention, there is provided directory assistance apparatus for use in a telephone system, comprising a voice processing unit having at least one lexicon of lexemes potentially recognizable by the unit and data representing a predetermined relationship between each of said lexemes and each of a plurality of call sources in an area served by the directory assistance apparatus. The unit also has means for issuing messages to a caller making a directory assistance call to prompt the caller to utter a required one of the lexemes, and means for detecting an identifier, for example a portion of a calling number, for the call source from whence the directory assistance call was received, means responsive to the identifier detected and to the data for computing a probability index for each lexeme representing the likelihood of that particular one of said lexemes being that uttered by the caller, and speech recognition means for selecting from the lexicon, on the basis of the acoustics of the caller's utterance and the probability index, a lexeme corresponding to that uttered by the caller.

A lexeme is a basic lexical unit of a language and comprises one or several words, the elements of which do not separately convey the meaning of the whole.

Preferably, the voice processing unit has several lexicons, each comprising a group of lexemes having a common characteristic, for example name, language, geographical area, and the speech recognition means accesses the lexicons selectively in dependence upon a previous user prompt.

Computation of the probability index may take account of a priori call distribution. A priori call distribution weights the speech recognition decision to take account of a predetermined likelihood of a particular locality containing a particular destination being sought by a caller. The apparatus may use the caller's number to identify the locality from which the caller is making the call.

The probability index might bias the selection in favour of, for example, addresses in the same geographical area, such as the same locality.

In preferred embodiments of the present invention the voice processing unit elicits a series of utterances by a subscriber and, in dependence upon a listing name being recognized, initiates automatic accessing of a database to determine a corresponding telephone number.

The apparatus may be arranged to transfer or "deflect" a directory assistance call to another directory assistance apparatus when it recognizes that the subscriber has uttered the name of a locality which is outside its own directory area. In such a situation, the above-mentioned predetermined relationship between the corresponding lexeme and the call source is that the lexeme relates to a locality which

is not served by the apparatus.

Thus, embodiments of the invention may comprise means for prompting a subscriber to specify a locality, means for detecting a place name uttered in response, means for comparing the uttered place name with the lexicon and in dependence upon the results of the comparison selecting a message and playing the message to the subscriber. If the place name has been identified precisely as a locality name served by the apparatus, the message may prompt the caller for more information. Alternatively, if the locality name is not in the area served by the apparatus, the message could be to the effect that the locality name spoken is in a different calling or directory area and include an offer to give the subscriber the directory assistance number to call. In that case, the speech recognition system would be capable of detecting a positive answer and supplying the appropriate area code. Another variation is that the customer could be asked if the call should be transferred to the directory assistance operator in the appropriate area. If the subscriber answered in the affirmative, the system would initiate the call transfer.

The caller's responses to the speech recognition system may be recorded. If the system disposed of the call entirely without the assistance of the operator, the recording could be erased immediately. On the other hand, if the call cannot be handed entirely automatically, at the point at which the call is handed over to the operator, the recording of selected segments of the subscriber's utterances could be played back to the operator. Of course, the recording could be compressed using the prior art techniques mentioned above.

According to a second aspect of the invention, a method of at least partially automating directory assistance in a telephone system using directory assistance apparatus comprising a voice processing unit having a lexicon of lexemes potentially recognizable by the unit and data representing a predetermined relationship between each of the lexemes and a calling number in an area served by the automated directory assistance apparatus, comprises the steps of:

issuing messages to a caller making a directory assistance call to prompt the caller to utter one or more utterances, detecting an identifier, such as a calling number originating a directory assistance call, computing, in response to the identifier and said data, a probability index for each lexeme representing the likelihood that the lexeme will be selected, and employing speech recognition means to recognize, on the basis of the acoustics of the caller's utterance and the probability index, a lexeme corresponding to that uttered by the caller.

Preferably, the voice processing unit has several lexicons, each having lexemes grouped according to certain characteristics e.g. names, localities, languages and the method includes the steps of issuing a series of messages and limiting the recognition process to a different one of the lexicons according to the most recent message.

The various objects, features, aspects and advantages of the present invention will become more apparent from the following detailed description, in conjunction with the accompanying drawings, of a preferred embodiment of the invention.

BRIEF DESCRIPTION OF DRAWINGS

FIG. 1 is a general block diagram of a known telecommunications system;

FIG. 2 is a simplified block diagram of parts of a telecommunications system employing an embodiment of the present invention;

FIGS. 3A and 3B are a general flow chart illustrating the processing of a directory assistance call in the system of FIG. 2;

FIG. 4 is a chart illustrating the frequency with which certain localities are requested by callers in the same or other localities; and

FIG. 5 is a graph of call distribution according to distance and normalized for population of the called locality.

BEST MODE FOR CARRYING OUT THE INVENTION

FIG. 1 is a block diagram of a telecommunications system as described in U.S. Pat. No. 4,797,910. As described therein, block 1 is a telecommunications switch operating under stored program control. Control 10 is a distributed control system operating under the control of a group of data and call processing programs to control various parts of the switch. Control 10 communicates via link 11 with voice and data switching network 12 capable of switching voice and/or data between inputs connected to the switching network. An automated voice processing unit 14 is connected to the switching network 12 and controlled by control 10. The automated voice processing unit receives input signals which may be either voice or dual tone multifrequency (DTMF) signals and is capable of determining whether or not the DTMF signals are allowable DTMF signals and initiating action appropriately. In the system described in U.S. Pat. No. 4,797,910, the voice processing unit is capable of distinguishing between the various elements of a predetermined list of spoken responses. The voice processing unit 14 also is capable of generating tones and voice messages to prompt a customer to speak or key information into the system for subsequent recognition by the speech recognition system. In addition, the voice processing unit 14 is capable of recording a short customer response for subsequent playback to a human operator. The voice processing unit 14 generates an output data signal, representing the result of the voice processing. This output data signal is sent to control 10 and used as an input to the program for controlling establishment of connections in switching network 12 and for generating displays for operator position 24 coupled to the network 12 via line 26. In order to set up operator assistance calls, switch 1 uses two types of database system. Local database 16 is directly accessible by control 10 via switching network 12. Remote database system 20 is accessible to control 10 via switching network 12 and interconnecting data network 18. A remote database system is typically used for storing data that is shared by many switches. For example, a remote database system might store data pertaining to customers for a region. The particular remote database system 20 that is accessed via data network 18 would be selected to be the remote database associated with the region of the called terminal. Interconnecting data network 18 can be any well known data network and specifically could be a common channel signalling system such as the international standard telecommunications signalling system CCS 7.

A transaction recorder 22, connected to control 10, is used for recording data about calls for subsequent processing. Typically, such data is billing data. The transaction recorder 22 is also used for recording traffic data in order to engineer additions properly and in order to control traffic dynamically.

The present invention will be employed in a telecommunications system which is generally similar to that described in U.S. Pat. No. 4,797,910. FIG. 2 is a simplified block diagram of parts of the system involved in a directory assistance call, corresponding parts having the same reference numbers in both FIG. 1 and FIG. 2. As shown in FIG.

2, block 1 represents a telecommunications switch operating under stored program control provided by a distributed control system operating under the control of a group of data and call processing programs to control various parts of the switch. The switch 1 comprises a voice and data switching network 12 capable of switching voice and/or data between inputs and outputs of the switching network. As an example, FIG. 1 shows a trunk circuit 31 connected to an input of the network 12. A caller's station apparatus or terminal 40 is connected to the trunk circuit 31 by way of network routing/switching circuitry 30 and an end office 33. The directory number of the calling terminal, identified, for example, by automatic number identification, is transmitted from the end office switch 33 connecting the calling terminal 40 to switch 1.

An operator position controller 23 connects a plurality of operator positions 24 to the switch network 12. Each operator position 24 comprises a terminal which is used by an operator to control operator assistance calls. Data displays for the terminal are generated by controller 23. A data/voice link 27 connects an automated voice processing unit 14A to the switching network 12. The automated voice processing unit 14A will be similar to that described in US patent number 4,797,910 in that it is capable of generating tones and voice messages to prompt a customer to speak or key dual tone multifrequency (DTMF) information into the system, determining whether or not the DTMF signals are allowable DTMF signals, initiating action appropriately and applying speech recognition to spoken inputs. In addition, the voice processing unit 14A is capable of recording a short customer response for subsequent playback to a human operator.

Whereas, in U.S. Pat. No. 4,797,910, however, the voice processing unit 14 merely is capable of distinguishing between various elements of a very limited list of spoken responses to determine the class of the call and to which operator it should be directed, voice processing unit 14A of FIG. 2 is augmented with software enabling it to handle a major part, and in some cases all, of a directory assistance call.

In order to provide the enhanced capabilities needed to automate directory assistance calls, at least partially, the voice processing unit 14A will employ flexible vocabulary recognition technology and a priori probabilities. For details of a suitable flexible vocabulary recognition system the reader is directed to Canadian patent application number 2,069,675 filed May 27, 1992 and laid open to the public Apr. 9, 1993, the entire disclosure of which is incorporated herein by reference.

A priori probability uses the calling number to determine a probability index which will be used to weight the speech recognition result. The manner in which the a priori probabilities are determined will be described in more detail later with reference to FIGS. 4 and 5.

While it would be possible to use a single lexicon to hold all of the lexemes which it is capable of recognizing, the voice processing unit 14A has several lexicons, for example, a language lexicon, a locality lexicon, a YES/NO lexicon and a business name lexicon. Hence, each lexicon comprises a group of lexemes having common characteristics and the voice processing unit 14A will use a different lexicon depending upon the state of progress of the call, particularly the prompt it has just issued to the caller.

As shown in FIGS. 3A and 3B, in embodiments of the present invention, when the voice processing unit 14A receives a directory assistance call, it determines in step 301 whether or not the number of the calling party is known. If it is not, the voice processing unit immediately redirects the call for handling by a human operator in step 302. If the

calling number is known, in step 303 the voice processing unit 14A issues a bilingual greeting message to prompt the caller for the preferred language and compares the reply with a lexicon of languages. At the same time, the message may let the caller know that the service is automated, which may help to set the caller in the right frame of mind. Identification of language choice at the outset determines the language to be used throughout the subsequent process, eliminating the need for bilingual prompts throughout the discourse and allowing the use of a less complex speech recognition system. If no supported language name is uttered, or the answer is unrecognizable, the voice processing unit 14A hands off the call to a human operator in step 304 and plays back to the operator whatever response the caller made in answer to the prompt for language selection. It will be appreciated that the voice processing unit 14A records the caller's utterances for subsequent playback to the operator, as required.

If the caller selects French or English, in step 305 the voice processing unit 14A uses the calling number to set a priori probabilities to determine the likelihood of the locality names in the voice processing unit's locality lexicon being requested. The locality lexicon comprises the names of localities it can recognize, as well as a listing of latitudes and longitudes for determining geographical distances between localities and calling numbers. In step 305, the voice processing unit 14A computes a priori probabilities for each lexeme in the locality lexicon based upon (i) the population of the locality corresponding to the lexeme; (ii) the distance between that locality and the calling number; and (iii) whether or not the calling number is within that locality. The manner in which these a priori probabilities can be determined will be described more fully later.

In step 306, the voice processing unit 14A issues the message "For what city?" to prompt the caller to state the name of a locality, and tries to recognize the name from its locality lexicon using speech recognition based upon the acoustics, as described in the afore-mentioned Canadian patent application number 2,069,675. The voice processing unit will also use the a priori probabilities to influence or weight the recognition process. If the locality name cannot be recognized, decision steps 307 and 308 cause a message to be played, in step 309, to prompt the caller for clarification. The actual message will depend upon the reason for the lack of recognition. For example, the caller might be asked to speak more clearly. Decision step 308 permits a limited number of such attempts at clarification before handing the call off to a human operator in step 310. The number of attempts will be determined so as to avoid exhausting the caller's patience.

If the locality name is recognized, the voice processing unit 14A determines in step 311 whether or not the locality is served by the directory assistance office handling the call. If it is not, the voice processing unit will play a "deflection" message in step 12 inviting the caller to call directory assistance for that area. It is envisaged that, in some embodiments of the invention, the deflection message might also give the area code for that locality and even ask the caller if the call should be transferred. It should be appreciated that, although some localities for other areas are in the lexicon, and hence recognizable, there is no corresponding data relating them to the calling numbers served by the apparatus since the apparatus cannot connect to them. The "predetermined relationship" between the localities for other areas and the calling numbers is simply that they are not available through the automated directory assistance apparatus which serves the calling numbers.

If the requested locality is served by the directory assistance office handling the call, in step 313 the voice processing unit will transmit a message asking the caller to state whether or not the desired listing is a business listing and employ speech recognition and a YES/NO lexicon to recognize the caller's response. If the response cannot be recognized, decision steps 314 and 315 and step 316 will cause a message to be played to seek clarification. If a predetermined number of attempts at clarification have failed to elicit a recognizable response, decision step 315 and step 317 hand the call off to a human operator. If a response is recognized in step 314, decision step 318 (FIG. 38) determines whether or not a business was selected. If not, step 319 plays the message "For what listing?" and, once the caller's response has been recorded, hands off to the human operator.

If decision step 318 indicates that the required number is a business listing, in step 320 the voice processing unit 14A plays a message "For what business name?" and employs speech recognition and a business name lexicon to recognize the business name spoken by the caller in reply. Once again, the recognition process involves an acoustic determination based upon the acoustics of the response and a priori probabilities.

If the business name cannot be recognized, in steps 321, 322 and 323 the unit prompts the caller for clarification and, as before, hands off to a human operator in step 324 if a predetermined number of attempts at clarification fail.

It should be noted that, when the unit hands off to a human operator in step 310, 317, 319 or 324, the operator's screen will display whatever data the automatic system has managed to determine from the caller and the recording of the caller's responses will be replayed.

If the unit recognizes the business name spoken by the caller, in step 325 the unit determines whether or not the database 16 lists a main number for the business. If not, the unit hands off to the human operator in step 326 and language, locality and selected business will be displayed on the operator's screen. If there is a main number for the business, in step 327 the unit plays a message asking if the caller wants the main number and uses speech recognition to determine the answer. If the caller's response is negative, step 328 hands off to the human operator. If the caller asks for the main number, however, in step 329 the unit instructs the playing back of the main number to the caller, and terminates the interaction with the caller.

As mentioned earlier, the use of a priori probabilities enhances the speech recognition capabilities of the voice processing unit 14A. Determination of a priori probabilities for locality names (step 305) will now be described. (A priori probabilities for other lexicons can be determined in a similar manner appropriate to the "predetermined relationship" for that lexicon.)

Statistics collected from directory assistance data show a relation between the origin of a call and its destination. An a priori model of probability that a person in a particular numbering plan area NPA and exchange NXX, i.e. with a phone number (NPA)NXX . . . , will ask for a destination locality d, is an additional piece of information which improves the recognition performance. The a priori model expresses the probability $P(d|o)$ of someone requesting a destination locality d given that they are making the directory assistance call from an originating locality o. The probability $p(d|o)$ depends upon the population of destination locality d and the distance between destination locality d and originating locality o. The originating locality o from

which the directory assistance call originates is not known precisely. From the originating phone number (NPA)NXX . . . , the Central Office (CO) is identified using the Bellcore mapping. Following that step, a set of possible originating localities associated with that Central Office is considered. The probability of a caller requesting directory assistance for a destination locality d from a phone number (NPA)NXX . . . in an originating locality o is:

$$P(d|(NPA)NXX) = \sum_{o \in CO} P(o)P(d|o) \quad (\text{Eq 1})$$

The probability $P(o)$ of each originating locality o associated with a CO originating a call is proportional to its population. Finally, the total recognition score for each locality is a weighted sum of the usual acoustic likelihood $\log P(Y_1 Y_2 \dots Y_N | d)$ and the logarithm of $P(d|o)$ calling (NPA)NXX:

$$\text{Score}(d) = \log P(Y_1 Y_2 \dots Y_N | d) + \lambda \log P(d|o) \text{ calling (NPA)NXX} \quad (\text{Eq 2})$$

An a priori model may be arranged to distinguish between populations having French or English as their first language. Knowing the language selected by the user, the population using that language is used to estimate $P(d|o)$. A minimum value of 10% of the population is used to avoid excessively penalizing a language.

An example of an a priori probability model developed using directory assistance data collected in the 514, 418 and 819 area codes is shown graphically in FIG. 4. In each of these area codes, the number of requests to and from each exchange (NXX) was collected; faint lines appear indicating the frequency of "any city requesting Montreal"; "Montreal requesting any city"; and "any city requesting itself". From these data it was possible to estimate the parameters of a parametric model predicting the probability of a request for information being made for any target locality given the calling (NPA)NXX. The parameters of the model proposed are the destination locality's population and the distance between the two localities. Where o is the originating locality, d is the destination locality, and S is the size of a locality, then the likelihood of a request about locality d given locality o is

$$L(d|o) = S(d) * f(\dots)$$

The normalized likelihood is

$$\bar{L}(d|o) = 0.60 \frac{L(d|o)}{\sum_{\text{over all } d'} L(d'|o)}$$

where d' denotes supported localities.

When the destination locality is also the originating locality, the likelihood is higher, so this is treated as a special case.

It is assumed that 60% of directory assistance (DA) requests are placed to localities including the originating locality as governed by the equation above, and an additional 40% of DA requests are for the originating locality, giving

$$\begin{aligned} P(d|o) &= \bar{L}(d|o), & d \neq o \\ &= \bar{L}(d|o) + 0.40, & d = o \end{aligned}$$

Intuitively, the probability $P(d|o)$ is a function $f(o, d)$ which

varies inversely with the distance between localities. In order to better define the function, a table of discrete values for certain distance ranges was derived from community of interest data collected in the three Quebec area codes. The distance units used in this section are the ones used by Bellcore to maintain geographical locality coordinates in North America. One kilometre is roughly equal to 1.83 Bellcore units. The discrete function values f computed for a given distance range in the province of Quebec are given in the Table below for each area code. Since the goal was to obtain an a priori model for the entire province, the values for $f(o,d)$ were computed for the province as a whole through factoring in the probability of a call originating in each area code. This was estimated to be in proportion to the number of exchanges [(NPA)NXX] per area code [NPA] relative to the province as a whole.

This gave $F\{\text{Province}\} = \{0.40f(514)\} + \{0.27f(819)\} + \{0.33f(418)\}$

distance	514	819	418	Province
0-25	1.0	1.0	1.0	1.00
26-50	0.9	0.3	0.7	0.67
51-75	0.4	0.0	0.2	0.23
76-100	0.1	0.0	0.3	0.14
101-125	0.1	0.0	0.1	0.07
126-150	0.1	0.0	0.1	0.07
151-175	0.0	0.0	0.2	0.07
176-200	0.0	0.0	0.0	0.00
>200	0.0	0.0	0.0	0.00

Given the sparseness of data, the model for obtaining weights as a function of distance was converted from nonparametric to parametric. For this purpose, a least square fit was performed on the data from ranges 26-50 to 151-175. The distance value used in the fitting process was the median distance in each range. An analysis of various function forms for the regression showed that the form below provided the closest fit to the collected data:

$$f(\text{distance}) = \{A/\text{distance}\} + B$$

The best coefficients obtained from the analysis were

$$A = 33.665$$

$$B = -0.305$$

This function f reaches zero when the distance is equal to 196 units. In order not to eliminate the possibility of handling a DA request when the distance was greater than this value, the function was modified to accommodate distances of up to twice the maximum distance between any pair of localities with population 10,000 or greater in the province. The two most distant localities that matched this criteria were RouynNoranda and Gaspé at a distance of 2,103 units. The maximum distance at which f would be zero was set to be 4,207 distance units. The function switches to a negative slope linear function at the point where the predicted value of f is 0.01. This corresponds to a distance value of 167.

The final f becomes

The fit of this model to the collected data, labelled "nonparametric model", is shown in FIG. 5.

In order to determine the effects of the a priori model on recognition rate, the model was applied to simulated DA requests, and each token in the test set was re-scored to take a priori likelihood into account. The function used for re-scoring was

$$\text{weighted score nas} = +K \log \{P(\text{old})\},$$

where nas is the normalized acoustic score, the acoustic score over the number of frames in the utterance. The proportionality constant K was trained to maximize the recognition rate over the province of Quebec. The distribution of tokens in the set used for development was normalized to be that predicted by the a priori model. For this reason a correctly recognised simulated DA request from a locality to the same locality carries more weight when computing recognition rate than does a request for a small distant locality with a correspondingly low a priori probability. A recognition rate was thus determined per locality and then the overall provincial recognition rate was computed by taking the sum of the rate for all localities in proportion to their respective populations. The only assumption made in applying the model was that the calling (NPA)NXX was known, which allowed the utterance to be re-scored by mapping it to all localities corresponding to the given entry in the Bellcore database.

The a priori model was further refined in order to avoid favouring the bigger localities unduly, as the recognition rate for these bigger localities, based on acoustics alone, was already above average. For this purpose, constants were introduced in the model corresponding to population ranges over the target localities in order to reduce the effective populations. These constants were not applied to the modelled distribution of requests since this would invalidate the method for computing the provincial recognition rate. The function defining likelihood becomes

$$L(d|o) = K_{r(d)} S(d) f(l)$$

where $r(d)$ is a range of destination locality population for which the constant K applies. The best ranges and their associated constants were then determined empirically from a development set.

Thus, using a priori call distribution, and flexible vocabulary recognition, embodiments of the present invention are capable of automating at least the front end of a directory assistance call and in some cases the entire call.

The embodiment of the invention described above is by way of example only. Various modifications of, and alternatives to, its features are possible without departing from the scope of the present invention. For example, the voice processing unit might be unilingual or multilingual rather than bilingual.

The probability index need not be geographical, but might be determined in other ways, such as temporal, perhaps according to time-of-day, or week or season-of-year. For example, certain businesses, such as banks, are unlikely to be called at one o'clock in the morning whereas taxi firms are. Likewise, people might call a ski resort in winter but not in summer. Hence the nature of the business can be used to weight the selection of certain lexemes for a particular enquiry.

The use of several lexicons, each comprising a group of lexemes, allows the voice processing unit to restrict the field of search for each prompt, thereby improving speed and accuracy. Nevertheless, it would be possible to use a single lexicon rather than the several lexicons described above.

It should be appreciated that although (NPA)NXX numbers have been mentioned, the invention is not limited to North American calls but with suitable modification could handle international calls.

It is envisaged that the voice processing unit 14A might dispense with computing a probability index for all destination localities served by the directory assistance apparatus. Instead, the locality lexemes could be grouped into predetermined subsets according to call identifiers, and the acoustic determination using speech recognition could sim-

7. Apparatus as claimed in claim 1, wherein the voice processing unit has one or more additional lexicons, each lexicon comprising a group of lexemes having a common characteristic, the computing means computes said index for

15. A method as claimed in claim 10, wherein the voice processing unit has one or more additional lexicons, each lexicon comprising a group of lexemes having a common

characteristic and the speech recognition means is employed to access the plurality of lexicons selectively in dependence upon one or more messages previously issued to the caller.

16. A method as claimed in claim 10, wherein the voice processing unit has one or more additional lexicons, each 5
lexicons comprising a group of lexemes having a common
characteristic, the computing means computes said index for
lexemes in the different lexicons selectively, in dependence
upon one or more messages previously issued to the caller
and the speech recognition means is employed to access the 10
plurality of lexicons selectively in dependence upon one or
more messages previously issued to the caller.

17. A method as claimed in claim 10, wherein the lexemes comprise names of businesses and the data comprise the nature of the business.

18. A method of at least partially automating directory assistance in a telephone system having directory assistance apparatus serving a predetermined area and comprising a voice processing unit having a lexicon of lexemes potentially recognizable by the unit, said lexemes including 20 lexemes corresponding to localities in a predetermined area served by the directory assistance apparatus and lexemes corresponding to localities not in the predetermined area, the method including the steps of:

using the voice processing unit to issue to a directory assistance caller a message inviting the caller to utter a name of a locality;

recognizing: one of said lexemes in the utterance;

determining whether or not the recognized lexeme is one of said lexemes corresponding to localities not in said predetermined area served by the apparatus; and

playing a message to the caller inviting the caller to direct the directory assistance request to a different directory assistance area in the event the recognized lexeme is not in the predetermined area.

19. Directory assistance apparatus, for a telephone system, comprising: a voice processing unit having at least one lexicon of lexemes potentially recognizable by the unit and data grouping the lexemes into predetermined subsets, each subset comprising lexemes preselected to give greater recognition accuracy for calls from a particular source; means for issuing messages to a caller making a directory assistance call to prompt the caller to utter one of said lexemes; means for detecting an identifier for the call source from whence the directory assistance call was received; means responsive to the detected identifier for selecting one of said predetermined subsets; and speech recognition means limited to the selected subset for recognizing, on the basis of the acoustics of the caller's utterance, a lexeme from said subset corresponding to that uttered by the caller.

15 20. A method of at least partially automating directory assistance in a telephone system in which directory assistance apparatus comprises a voice processing unit having a lexicon of lexemes potentially recognizable by the unit, and data grouping the lexemes into predetermined subsets, each
20 subset preselected as giving greater recognition accuracy for calls from a particular source, the method comprising the steps of:

issuing messages to a caller making a directory as a distance call to prompt the caller to utter one or more utterances:

detecting an identifier for a call source from whence the directory assistance call was received;

selecting on the basis of the identifier one of said predetermined subsets; and

employing speech recognition means to recognize, from the selected subset and on the basis of the acoustics of the caller's utterance, a lexeme corresponding to that uttered by the caller.

* * * * *

Requested Patent: EP0601710A2

Title:

ON DEMAND LANGUAGE INTERPRETATION IN A TELECOMMUNICATIONS
SYSTEM. ;

Abstracted Patent: EP0601710 ;

Publication Date: 1994-06-15 ;

Inventor(s):

DAVITT MICHAEL (US); DUNN ALAN N (US); GOLDSTEIN PAULA M (US);
GRIJALVA EDGAR J (US); NEAL MICHAEL (US); PETERSON CHRISTINE
PATRICIA (US); VAIOS CHRISTOS I (US) ;

Applicant(s): AMERICAN TELEPHONE TELEGRAPH (US) ;

Application Number: EP19930308840 19931104 ;

Priority Number(s): US19920973872 19921110 ;

IPC Classification: H04M3/42 ; H04Q3/00 ;

Equivalents: JP6225024, US5392343

ABSTRACT:

Easily accessed and widely available language interpretation services are provided in a public switched telephone network by a common platform adjunct which automatically connects an interpretation services subscriber with a selected language interpreter associated with a language interpretation platform in the network. A subscriber dials, for example, an international telephone number which includes a code indicating that the call is an international call, a country code, a city code, and a local destination number. The ANI of the subscriber is detected and the call is routed to the adjunct which further verifies and validates the subscriber. The adjunct places a call through the public switched telephone network to the language interpretation platform. The call is answered either by an automatically preselected interpreter or by a human operator who causes the call to be manually transferred to a desired interpreter. The international call is completed to the destination and the calling subscriber, the interpreter, and the called party are bridged together.



403433 "SEP 20 1994"



⑪ Publication number : **0 601 710 A2**

⑫ **EUROPEAN PATENT APPLICATION**

⑲ Application number : **93308840.3**

⑤① Int. Cl.⁵ : **H04M 3/42, H04Q 3/00**

⑳ Date of filing : **04.11.93**

③① Priority : **10.11.92 US 973872**

④③ Date of publication of application :
15.06.94 Bulletin 94/24

⑧④ Designated Contracting States :
DE FR GB

⑦① Applicant : **AMERICAN TELEPHONE AND
TELEGRAPH COMPANY**
32 Avenue of the Americas
New York, NY 10013-2412 (US)

⑦② Inventor : **Davitt, Michael**
50 Hamilton Avenue
Berkley Heights, New Jersey 07922 (US)
Inventor : **Dunn, Alan N.**
69 Lawrence Road
Randolph, New Jersey 07869 (US)

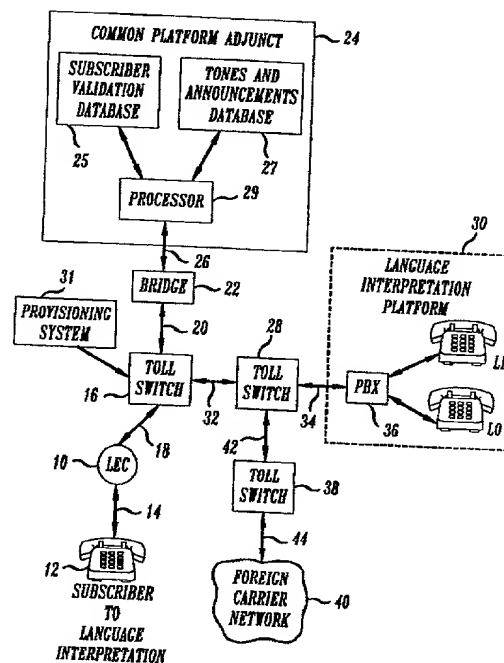
Inventor : **Goldstein, Paula M.**
15 Oaklyn Lane
Aberdeen, New Jersey 07747 (US)
Inventor : **Grijalva, Edgar J.**
1614 Union Turnpike 3c
North Bergen, New Jersey 07047 (US)
Inventor : **Neal, Michael**
12 Chestnut Place
Lebanon, New Jersey 08833 (US)
Inventor : **Peterson, Christine Patricia**
121 Grant avenue
Fords, New Jersey 08863 (US)
Inventor : **Vaios, Christos I.**
84 Garden Road
Shrewsbury, New Jersey 07702 (US)

⑦④ Representative : **Watts, Christopher Malcolm**
Kelway, Dr. et al
AT&T (UK) Ltd.
5, Morningson Road
Woodford Green Essex, IG8 0TU (GB)

⑤④ **On demand language interpretation in a telecommunications system.**

⑤⑦ Easily accessed and widely available language interpretation services are provided in a public switched telephone network by a common platform adjunct which automatically connects an interpretation services subscriber with a selected language interpreter associated with a language interpretation platform in the network. A subscriber dials, for example, an international telephone number which includes a code indicating that the call is an international call, a country code, a city code, and a local destination number. The ANI of the subscriber is detected and the call is routed to the adjunct which further verifies and validates the subscriber. The adjunct places a call through the public switched telephone network to the language interpretation platform. The call is answered either by an automatically preselected interpreter or by a human operator who causes the call to be manually transferred to a desired interpreter. The international call is completed to the destination and the calling subscriber, the interpreter, and the called party are bridged together.

FIG. 1



Technical Field

This invention relates to telecommunications between parties who speak different languages. More specifically, this invention relates to language interpretation services provided in a telecommunications system.

Background of the Invention

Today's telecommunication systems make it routine for persons from different countries to communicate with each other on a regular basis. Very often, however, the parties to such a call do not speak the same language. The usefulness of today's telecommunications systems could be greatly improved if there were some way to translate communications from one language to another in a telephone network.

There have been proposals to develop computers in telecommunication systems which can automatically translate voice communications from one language into another language. These efforts are currently in a somewhat rudimentary stage and are far from becoming a commercially practical reality.

In the meantime, AT&T offers a language interpretation service which is currently a part of the AT&T switched network. Known as the AT&T Language Line® Service; it allows a caller to contact a human interpreter for assistance in making a telephone call expected to involve parties speaking different languages. The caller dials an 800 number to reach the service after which the caller gives his or her credit card or AT&T calling card number to an operator. An operator takes additional information from the caller about the call including the phone number of the called party and the languages expected to be spoken. The operator then connects the caller to a human interpreter fluent in the languages to be spoken during the phone call. The caller or operator completes the call to the called party resulting in a conference call between the caller, the called party, and the interpreter.

Summary of the Invention

Although AT&T's Language Line® Service is a distinct and commercially significant improvement in the way telecommunications services are provided in today's networks, we have discovered that interpretation services provided by a telecommunications network can be markedly improved if the interpretation service were easier to access and if more people were able to partake of the service.

In this regard, there are two aspects of existing language interpretation services in telecommunications systems which can be improved. First, the amount and complexity of the information which must be entered into the telecommunications system to ini-

tiate and set up a phone call using the interpretation service is too great for convenient use of the service. Second, the requirement that only callers having credit cards or phone cards may use the service unduly limits the number of potential customers for the service. This invention provides a language interpretation service which is easier to use than existing language interpretation services. In particular, language interpretation is automatically made available for telephone calls initiated by conventional direct dialing procedures or by substantially similar procedures normally used to make a telephone call to a desired destination. No extra telephone numbers need to be used and billing for the services may be accomplished without a need for a credit card or a phone card.

In one example of the invention, a presubscribed caller is automatically given access to an interpretation service whenever a standard telephone call is made from a directory number associated with the subscriber stored in the network. No special 800 number must be dialed and the caller need not have a credit card or phone calling card. In another example of the invention, a caller may be given automatic access to an interpretation service by dialing a special prefix or suffix along with the direct dial telephone number of a called party.

There are just two examples of the invention, the full scope of which is defined in the claims appended to this application. Other examples of the invention will be apparent from the claims and the following detailed description of the preferred embodiments.

Brief Description of the Drawing

FIG. 1 is a block diagram of a public switched telephone network providing a language interpretation service in accordance with this invention.

FIGS. 2A to 2D depict a flow chart representing an illustrative call flow for a language interpretation service provided by the network of FIG. 1.

Detailed Description

FIG. 1 shows an example of a public switched telephone network architecture which may be used to implement various examples of this invention. FIG. 1 includes a schematic diagram of representative portions of a typical telecommunications network which includes a long distance public switched telephone network provided by a long distance carrier such as AT&T. As shown in FIG. 1, entry into the long distance network is from a local public switched telephone network provided by a local exchange carrier (LEC) such as one of the Regional Bell Operating Companies. In this example, the long distance public switched telephone network implements a language interpretation service in accordance with the invention, as described in more detail below. The language interpretation

service could also be provided in one or more of the local public switched telephone networks such as those by which telephone customers typically gain access to long distance telephone networks.

The network shown in FIG. 1 includes a public switched telephone network 10 which provides local telephone service to a number of telephone customers. One of those customers is a language interpretation service subscriber 12 shown in FIG. 1. The subscriber 12 is connected to the network 10 by a suitable subscriber line 14 which may provide a voice circuit and suitable signaling capability, such as dual tone multiple frequency (DTMF) signaling capability. The network 10 is connected to an originating toll switch 16 in the long distance network by means of a suitable trunk connection 18.

A line 20, which may be configured to operate as a primary rate interface (PRI), connects the switch 16 to a teleconference bridge 22 which will be used to create a conference between the subscriber 12, an interpreter, and a called party. The bridge 22 is connected to a common platform adjunct 24 by means of a suitable line 26. The adjunct 24 is a computer which effectuates the routing of calls through the network to connect the subscriber 12 with a called party and a language interpretation system in the network. The adjunct 24 may contain a subscriber validation database 25 containing profiles of those who subscribe to the interpretation service. The adjunct 24 also may contain a tones and announcements database 27 containing announcements and tones by which the adjunct may send appropriate messages to the subscriber 12. The adjunct 24 may also contain a processor 29 which may have a voice recognition circuit which receives and act upon voice responses from the subscriber. The processor 29 may also have detection circuitry which receives and acts upon signaling received from the subscriber 12. The processor 29 routes telephone calls through various parts of the network, and also selectively retrieves information from the database 25, selectively presents the announcements and tones from the database 27, and otherwise coordinates the activities of the adjunct 24 with the activities of the rest of the public switched network via the bridge 22. Although the adjunct 24 and bridge 22 are shown in FIG. 1 to be connected to the originating switch 16, those elements may be connected to any convenient switch in the network. In addition, although the bridge 22 is shown as a component separate from the adjunct 24 and the switch 16, it may also be implemented as a hardware or software structure in either the adjunct 24 or the switch 16.

The toll switch 16 is trunked to a toll switch 28 which acts as a terminating switch for a language interpretation platform 30. The switch 28 is connected to the switch 16 by means of a trunk connection 32. Although a single direct trunk connection 32 is shown

in FIG. 1, there may be additional switches and other network circuit elements between the switch 16 and the switch 28 depending on the locations of the subscriber 12, the adjunct 24, the platform 30, and the called party.

The toll switch 28 is connected to the language interpretation platform 30 by means of a line 34 which may be a primary rate interface. A PBX 36 in the platform 30 is connected to line 34 and serves to route the calls to an appropriate language interpreter LI. The PBX may be connected to any number of language interpreters only one of which is shown in FIG. 1. The PBX 36, in some examples of the invention, may be connected to one or more live operators LO who may facilitate the connection of the subscriber 12 to an appropriate language interpreter LI. The interpreters LI could reside at the PBX 36 as shown in FIG. 1 or they could reside at any other PBX or off a local switch in the network.

The toll switch 28 is connected to a toll switch 38 which acts as an international gateway between the domestic long distance network shown in FIG. 1 and one or more telephone networks operated by foreign carriers, one of those networks being shown in FIG. 1 and given reference numeral 40. The toll switch 28 is connected to the international gateway toll switch 38 by means of a suitable trunk connection 42 which may or may not have additional switches and network elements between the toll switch 28 and the toll switch 38 depending on the relative locations of the switch 28, the platform 30 and the toll switch 38. The switch 38 is connected with the foreign network 40 via suitable transmission equipment 44.

The language interpretation features of the telephone network shown in FIG. 1 allow subscribers to receive language interpretation support for all direct dialed international voice calls, for example. Subscribers may place international voice calls using current international plain old telephone service (POTS) dialing procedures. In one example of the invention, users will presubscribe to the interpretation service. In other examples of the invention, end users will not have to presubscribe. In some examples of the invention, subscribers may initiate calls involving language interpretation from their own directory number or from other predefined telephone numbers such as those associated with certain public phones in airports or other transportation facilities. In addition, interpretation services may be obtained not only for international telephone calls, but also for domestic telephone calls known to involve parties speaking different languages, including toll-free domestic telephone calls.

In one detailed example of the invention, shown in FIGs. 2A-2B, subscribers to language interpretation services will first dial a designation that the call will be an international call. For example, subscribers will first dial 011. Next, the subscriber will dial a coun-

try code indicating the country to which the phone call is to be directed and a national number representing the destination number of the called party. The numbers dialed by the subscriber are passed through the local telephone network 10 and are received by the toll switch 16 in block 46. The toll switch 16 also receives and screens the subscriber's automated number identification (ANI) in block 48. In block 50, the switch 16 checks to see if this call is coming from a directory number which is stored in the switch 16 as having subscribed to the interpretation service. In this regard, the switch 16 is programmed by a provisioning system 31 with a list of telephone numbers representing those which are associated with subscribers to the interpretation service. If the switch 16 determines that the call is being made from a telephone number which is subscribed to the interpretation service, and the caller is thereby identified as an interpretation service subscriber, the call is routed by the switch 16 in block 52 to the adjunct 24. If the ANI is not identified as belonging to a subscriber, then the call is routed normally, as shown in block 54. The adjunct 24 will further validate the ANI in block 56 by checking suitable information stored in database 25. For example, the adjunct 24 may check a list of subscribers who have past due and unpaid bills. In some examples of the invention, entry of a password known only to the subscriber may be required. The adjunct 24 will check to see if the entered password is a password associated with a subscriber. If the subscriber is not validated, service is denied or the call is not completed, or both, as shown in block 58. If the subscriber is validated, the adjunct 24 may prompt the user in block 60, for example, by playing an announcement, for further information on call handling. The prompt of the subscriber can be either a tone (e.g., a "bong") or an announcement. An example of a suitable announcement would be an indication that the caller has reached an on-demand interpretation service. The announcement may state that the caller should enter a predetermined shorthand activation code indicating that the caller wishes to use the interpretation service. For example, the announcement may request that the caller press a certain sequence of buttons on a Touch Tone telephone such as the * key followed by one of the numerical keys. A plurality of different interpretation services may be offered and the subscriber may indicate a selection of one of those services by the code which he or she enters. The announcement may also indicate to the subscriber that he or she may forego the interpretation service by entering a service refusal code, for example, by pressing the # symbols on a Touch Tone telephone. Following the tone or the announcement, a timer is set by the adjunct 24. If the timer expires, for example, after a period of five seconds from the tone or announcement, the call will be handed over from the adjunct 24 to the toll switch 16 to be routed like a normal

POTS call as shown in block 54 and the line between blocks 54 and 60 in FIG. 2A. If the subscriber signifies that he or she does not wish to use the interpretation service by entry of a service refusal code, the adjunct 24 will immediately hand the call over to the switch 16 for normal processing in block 54.

If the subscriber requests language interpretation by entry of the service activation code, the adjunct 24 will initiate a call request in block 62 to the PBX 36 in the language interpretation platform 30. To accomplish this, the adjunct 24 is preprovisioned with the destination number of the platform 30. The adjunct 24 sends a setup message to the PBX 36 including the directory number of the subscriber. The message may include the country code dialed by the subscriber and the national number of the called party. The phone call may be connected with a preselected language interpreter, as shown in block 64, in a number of different ways. The phone call from the adjunct 24 may be directed first to a live operator LO who may assist the caller in reaching a desired language interpreter. The country code alone or the combination of country code and city code may be used by the live operator LO to help determine which language interpreter should be used. In this regard, there may be circuitry in PBX 36 which produces a list of languages displayed to the operator and likely to be spoken in the geographical area represented by the country code and city code. Alternatively, the call from the adjunct 24 may be automatically directed to an appropriate language interpreter LI without the intervention of a human operator LO. In this case, the language interpretation platform may include circuitry which detects the nature of the country code, city code, or both the country and city codes, and automatically brings an appropriate language interpreter LI to the phone call based on the predominant languages spoken in the geographical area represented by the country code and city code.

The call to the PBX 36 is routed via the bridge 22, the toll switch 26, and the toll switch 28. Upon receipt of a connect message, the human operator or the automatically ascertained language interpreter are bridged onto the call. If a human operator is connected, he or she will assist the subscriber in finding the desired language interpreter and, if needed, will collect any other needed information from the subscriber. Once the subscriber is validated and the language interpreter becomes available, the call is then transferred to the selected interpreter who is now bridged onto the call. At this point, conversation can take place between the subscriber and the interpreter during which the role played by the interpreter in the upcoming phone conversation with the called party may be determined. At the same time, billing for the services of the interpretation system may begin at the PBX 36. The billing for the interpretation services will be added to the billing for the actual telephone call.

Since this service part of the call is billed by the PBX 36, the adjunct 24 must insure that the subscriber's destination number is delivered to the PBX 36 so that the bill for the interpretation service can appear on the regular bills relating to the caller's telephone. Alternatively, the switches or the adjunct 24 could produce a billing record.

When the two-way conversation between the subscriber and the interpreter is concluded and its agreed to complete the actual phone call to the called party, the subscriber or the language interpreter may trigger the adjunct 24 with a suitable signal which will cause the adjunct 24 to complete the international leg of the call setup, as shown in block 66 in FIG. 2B. The adjunct 24 will send, via the bridge 22, the subscriber dialed country code and national number to the toll switch 16 and toll switch 28 for further processing and routing to the international gateway switch 38 as an international POTS telephone call. At this point, bridging between the subscriber, the interpreter, and the switch 16 is considered to be activated, but waiting for cut-through to the international switch 38. When the country code and national number are received by the switch 16 from the adjunct 24, the rest of the switches between the switch 16 and the switch 38 will route call to the switch 38 and thereafter to the foreign carrier network 40. The set-up of the call is considered complete when a cut-through message is received. The ultimate result is that there is a conference call created by bridging the subscriber, the interpreter, and the called party, as shown in block 68. Once the conference call has been created in block 68, the adjunct 24 may receive a request from one of the conference participants to drop the interpreter from the conference in block 69 in FIG. 2C. The adjunct 24 then drops the part of the conference circuit connecting the language interpreter to the conference in block 70. The adjunct then hands the call over to the switch in block 71 and waits for a new call request in block 72. At the end of the phone call, the adjunct receives a request to drop all circuits involved in that particular call as shown in block 73 in FIG. 2D. The adjunct terminates all the circuit connections involved in the call in block 74 and waits for a new call request in block 75. Call disconnect will occur on receipt of an on-hook signal from a subscriber. This on-hook signal will trigger the adjunct 24 to release the connection to the PBX 36 and toll switches 16, 28 and 38. The interpreter is able to drop off from the call at any time without terminating the call. In addition, the call could be handed over to an appropriate switch in the network at any moment if one of the end users determine that there is no need for language interpretation. This handover to a switch can be initiated by the originator of the call, for example, by dialing the * key on a Touch Tone telephone.

The interpretation services provided by this invention may be obtained for any direct dialed interna-

tion calls originating from residential or other sources in any country. In addition to having this service available from a subscriber's own telephone number, the service may also be made available to subscribers from selected other locations, for example, from certain airport telephones and the like. The service may even be made available to certain non-subscribers who use toll-free services such as the international inbound services, known as 1-800 services. In these situations, the telephone numbers of these other phones will be screened in block 48 in the same manner that the telephone numbers of the subscribers are screened in the example of the invention described above. In addition to a list of acceptable telephone numbers from which interpretation services may be obtained, the adjunct 24 will also have a look-up table of subscriber codes associated with the normal directory numbers of the subscribers and individual passwords/I.D.'s. When a subscriber wishes to use the interpretation service from a telephone other than his or her normal telephone, the subscriber will be prompted in the course of performing the validation in block 56 to enter his or her normal destination number followed up a multi-digit access code. Upon successful validation by the adjunct 24, the call processing may proceed in the same manner as in the example of the invention described above.

In a menu-driven example of this invention, the subscriber dials the destination number of the phone call in block 46 and is given the previously described tone or announcement in block 60 relating to entry into the interpretation service. The subscriber then indicates a desire to enter the service as described above by entry of an activation code. In connection with performing the call connection of block 64, an automated voice system in the platform 30 answers and provides the subscriber with a menu of languages to choose from based on the country code and city code entered by the subscriber. For example, in a call to Strasbourg, France, the subscriber may be provided with the option of choosing either a French interpreter or a German interpreter. In a call to Switzerland, the option of choosing a French, German, or Italian interpreter may be given. Selection is made by pressing certain dialing codes on the subscriber's telephone. The subscriber may be given another code to select if he or she desires personal assistance from a human operator. Voice recognition circuitry may be provided in the language interpretation platform 30 so that a subscriber may make a selection of interpreter by voice command rather than by entry of codes into his or her telephone.

In yet another example of the invention, the subscriber may be permitted to enter an additional code after entry of the service activation code to indicate a selection of an interpreter dealing in a desired language. In this situation, the subscriber may be given the opportunity of reaching a human operator by di-

aling an additional code after entry of the activation code.

There may also be stored in a subscriber's profile in the database 25 that this subscriber normally wishes to use an interpreter fluent in a particular language. The subscriber will then be automatically connected to such interpreter each time the subscriber uses the interpretation services.

In an additional variation of the invention, a caller keys in an international telephone number and a special short hand designation, such as the designation produced by depressing the * key on a Touch Tone telephone. In one example, the special designation may be a prefix dialed before the international telephone number is dialed and in another example, the special designation may be a suffix dialed after the international number is dialed. The designation causes the caller to be automatically connected to an interpretation services in the network. The platform may detect the country code in the international telephone number and route the call to an interpreter who speaks English and the predominant language spoken in the country represented by the entered country code. The telephone number from which the call was initiated will be billed for the interpretation service and the international phone call. The interpreter may complete the phone call and remain bridge onto the call to provide language translation for the call as in other examples of the invention.

An interpretation service in accordance with this invention could be configured to route calls to interpreters residing in foreign networks. For example, calls could be routed to interpreters connected with PTT's or AT&T's Global Network. Collect calls could be handled by provision of a common platform adjunct having a database to validate the called party and play appropriate announcement to the called party.

Claims

1. A telecommunications apparatus, comprising:
means for receiving a telephone call comprising at least a telephone number of a caller and a telephone number of a called party; and
means for detecting a predetermined characteristic of the caller's telephone number and automatically directing the telephone call to an interpretation service.
2. A method of providing a language interpretation service in public switched telephone network, comprising the steps of:
receiving a telephone call comprising at least a caller's telephone number and a called party's telephone number, and
automatically providing an interpretation

service in response to a predetermined characteristic of the caller's telephone number.

3. The method of claim 2, in which the predetermined characteristic is whether or not the caller's telephone number is stored in the network as a telephone number associated with a subscriber to the interpretation service.
4. The method of claim 3, in which providing step comprises the step of automatically routing the telephone call to a common platform adjunct in response to a determination that the caller's telephone number is a telephone number associated with a subscriber to the interpretation service.
5. The method of claim 4, in which the providing step further comprises the step of validating the subscriber.
6. The method of claim 4, in which the providing step further comprises the step of sending a message to the subscriber giving the subscriber an option to activate the interpretation service.
7. The method of claim 6, in which the providing step further comprises the step of receiving a message from the subscriber indicating a desire to activate the interpretation service and automatically routing the telephone call to a language interpretation platform in the network.
8. The method of claim 7, in which the providing step further comprises the step of receiving the telephone call in the language interpretation platform and connecting the telephone call to a language interpreter desired by the subscriber.
9. The method of claim 8, in which the step of receiving the telephone call in the platform comprises the step of connecting the telephone call to a human operator and transferring the call from the human operator to a language interpreter desired by the subscriber.
10. The method of claim 8, in which the step of receiving the telephone call in the platform comprises the step of automatically connecting the telephone call to a desired language interpreter in response to a predetermined characteristic of the telephone number of the called party.
11. The method of claim 10, in which the predetermined characteristic of the telephone number of the called party is a country code.
12. The method of claim 2, in which the public switched telephone network is a long distance tele-

phone network.

13. The method of claim 2, in which the public switched telephone network is a local telephone network. 5
14. The method of claim 4, in which the providing step includes the step of routing the telephone call to a language interpreter desired by the caller. 10
15. The method of claim 14, in which the providing step further comprises the steps of:
 routing the telephone call to the called party; and
 bridging together the caller, the language interpreter, and the called party. 15
16. The method of claim 2, in which the telephone call is an international telephone call. 20
17. The method of claim 2, in which the telephone call is an domestic telephone call.
18. The method of claim 2, further comprising the step of: 25
 receiving an additional caller's telephone number, and
 in which the providing step includes the step of responding to the first mentioned caller's telephone number and the additional caller's telephone number to determine whether or not the first mentioned caller's telephone number is associated with a predetermined telephone from which language interpretation service may be obtained and to determine whether or not the additional caller's telephone number is associated with a subscriber to the language interpretation service. 30
 40
19. The method of claim 2, in which the providing step includes the step of sending a menu of choices to a caller relating to languages spoken by language interpreters available to assist the caller. 45
20. The method of claim 2, in which the providing step includes the step of automatically routing the telephone call to a language interpreter previously selected by a subscriber for automatic connection to the subscriber each time the subscriber uses the language interpretation services. 50
 55
21. A method of providing language interpretation in a public switched telephone network, comprising the steps of: 7

receiving a telephone call comprising a called party's telephone number and a short hand designation appended to the called party's telephone number representing a request for language interpretation; and

automatically providing language interpretation in response to the called party's telephone number and the short hand designation.

22. The method of claim 21, in which the short hand designation is a suffix appended to the called party's telephone number.

23. The method of claim 21, in which the short hand designation is a prefix appended to the called party's telephone number.

FIG. 1

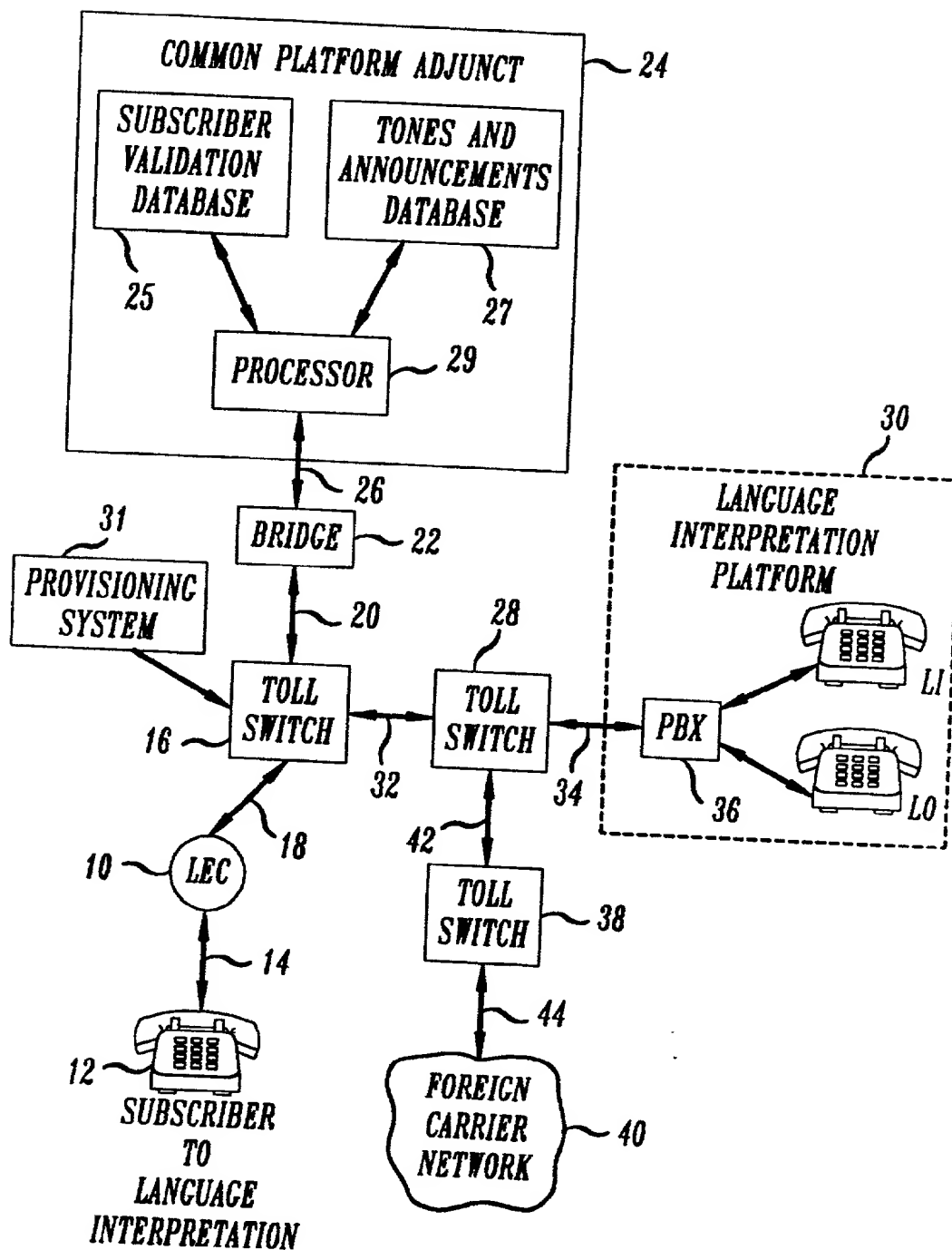


FIG. 2A

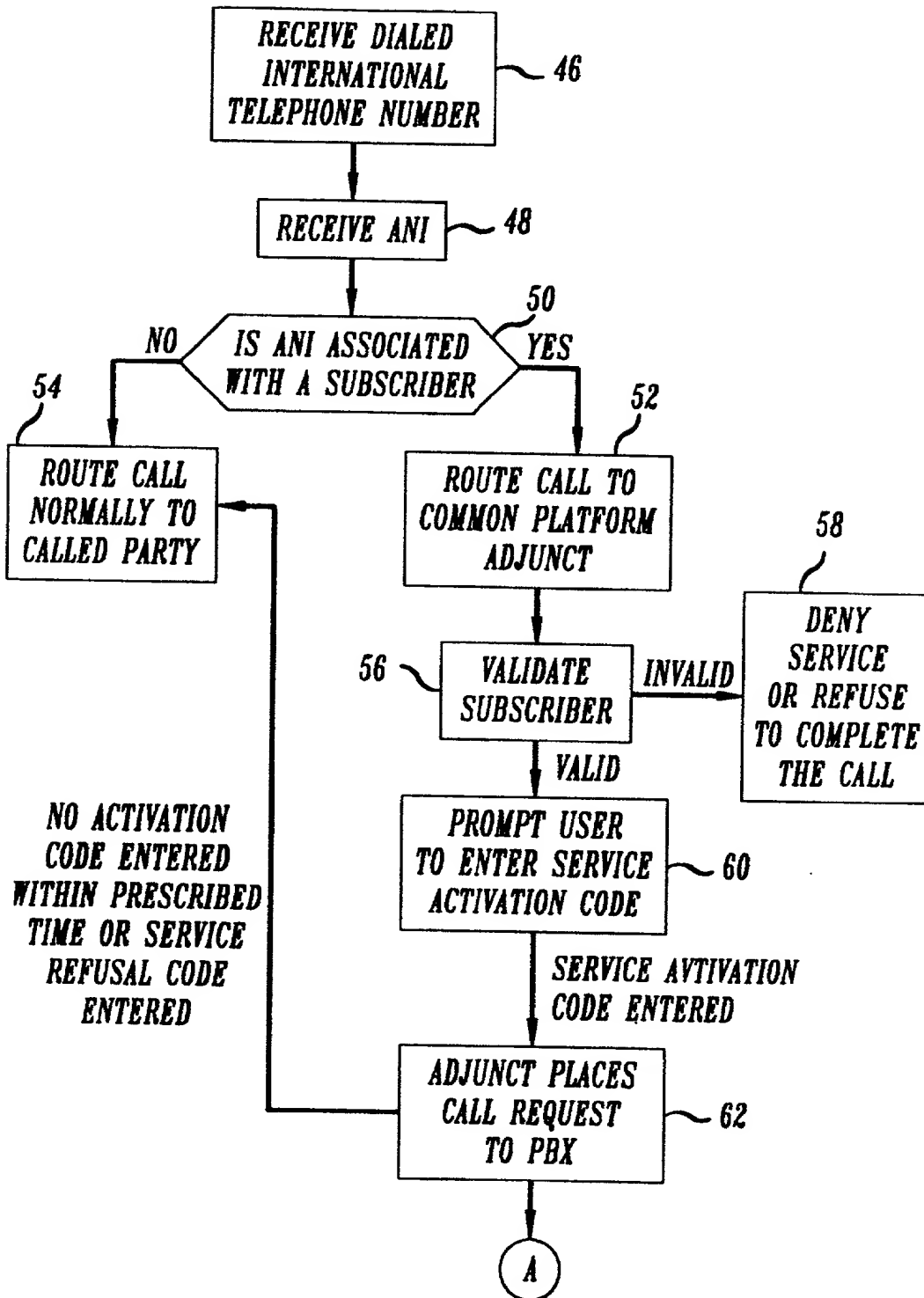


FIG. 2B

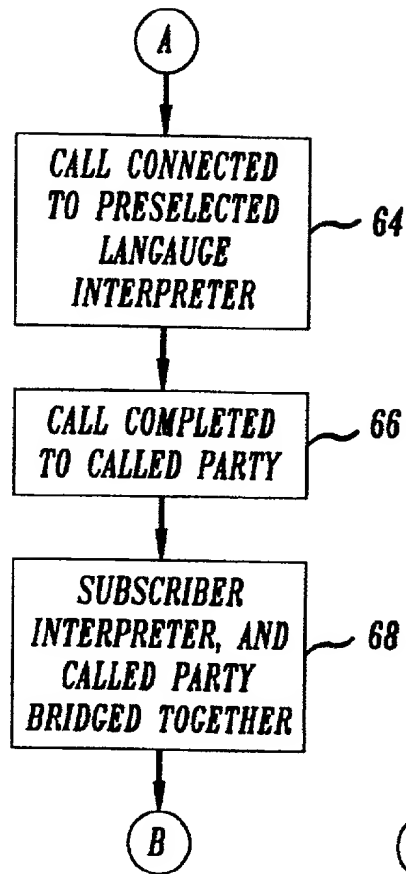


FIG. 2C

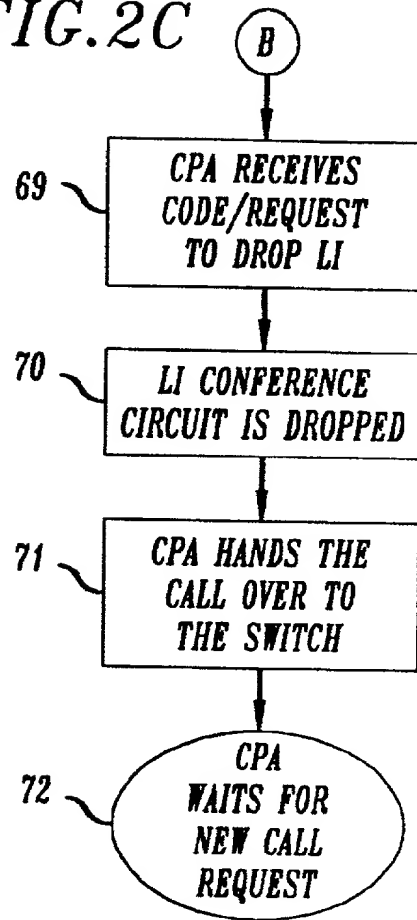
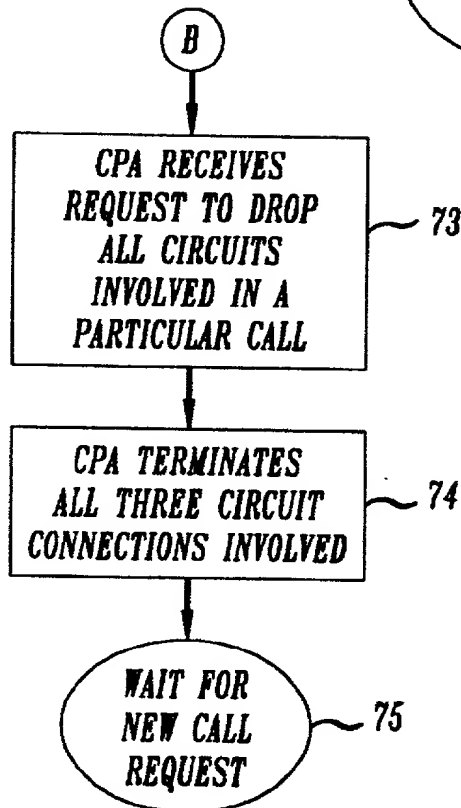


FIG. 2D



Requested Patent: EP0484070A2
Title: EDITING COMPRESSED VOICE INFORMATION. ;
Abstracted Patent: EP0484070 ;
Publication Date: 1992-05-06 ;
Inventor(s): PESSIA DARIO (FR); HOPPER ANDREW (US) ;
Applicant(s): IBM (US) ;
Application Number: EP19910309913 19911025 ;
Priority Number(s): US19900606798 19901030 ;
IPC Classification: G10L5/02 ; G10L5/04 ;
Equivalents: DE69131155D, JP2880592B2, JP4248598



ABSTRACT:

A method and apparatus for editing the displayed voice wave form by marking the portion of interest on the screen is disclosed. Marked segment may then be deleted, for example, or copied into another segment in second voice editing window. In either case, pointers are established at the selected marker positions of the displayed voice segment and in the corresponding positions of uncompressed voice segments. The voice data is treated as a stream of fixed-length micro-segments, where there is a predictable correlation between the positions of the compressed and uncompressed data. In the implementation at hand, these micro-segments are 20 ms. in length. Editing is accomplished by modifying micro-segments in both the compressed and uncompressed segments simultaneously. When the user is satisfied with the result, the edited wave form is redrawn on the screen. The user may then SAVE the result, and the entire segment is rewritten to the data base, replacing the previous version. Only the compressed version is written, thus eliminating the need for a subsequent pass through the compression hardware with the associated compounding of distortion.



(11) Publication number: **0 484 070 A2**

(12)

EUROPEAN PATENT APPLICATION

(21) Application number: **91309913.1**

(51) Int. Cl.⁵: **G10L 5/02**

(22) Date of filing: **25.10.91**

(30) Priority: **30.10.90 US 606798**

(43) Date of publication of application:
06.05.92 Bulletin 92/19

(84) Designated Contracting States:
DE FR GB

(71) Applicant: **International Business Machines Corporation**
Old Orchard Road
Armonk, N.Y. 10504 (US)

(72) Inventor: **Hopper, Andrew**
227 Homer Avenue
Palo Alto, CA 94301 (US)
Inventor: **Pessia, Dario**
421 Chemin de la Rourière
F-06610 La Gaude (FR)

(74) Representative: **Killgren, Neil Arthur**
IBM United Kingdom Limited Intellectual
Property Department Hursley Park
Winchester Hampshire SO21 2JN (GB)

(54) **Editing compressed voice information.**

(57) A method and apparatus for editing the displayed voice wave form by marking the portion of interest on the screen is disclosed. Marked segment may then be deleted, for example, or copied into another segment in second voice editing window. In either case, pointers are established at the selected marker positions of the displayed voice segment and in the corresponding positions of uncompressed voice segments. The voice data is treated as a stream of fixed-length micro-segments, where there is a predictable correlation between the positions of the compressed and uncompressed data. In the implementation at hand, these micro-segments are 20 ms. in length. Editing is accomplished by modifying micro-segments in both the compressed and uncompressed segments simultaneously. When the user is satisfied with the result, the edited wave form is redrawn on the screen. The user may then SAVE the result, and the entire segment is rewritten to the data base, replacing the previous version. Only the compressed version is written, thus eliminating the need for a subsequent pass through the compression hardware with the associated compounding of distortion.

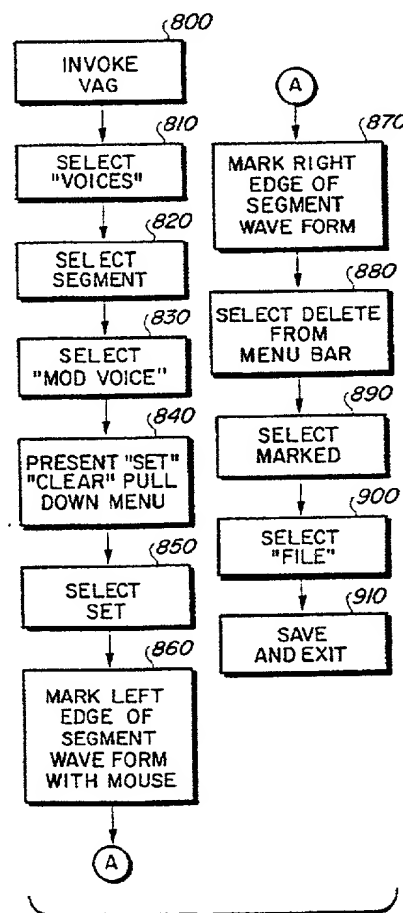


FIG. 16

EP 0 484 070 A2

This invention generally relates to improvements in voice messaging and more particularly to editing voice information without affecting the fidelity of the original voice information.

Information processing systems having a voice generating capability are presently employed as answering machines, voice messaging systems, voice response units and in general as intelligent peripherals. The voice signal may be prerecorded on audio tape or may be digitized, compressed and stored, for example, on a magnetic disk.

A typical application couples the information processing system to one or more phone lines, the system detecting the occurrence of a ring signal and answering the phone. Often a standard prompt voice message is sent to the phone line. Depending on the type of system the caller may depress certain buttons on a Touch-Tone phone set in order to inform the system of a specific type of action desired by the user. For example, after hearing the message, the information processing system may have access to a large data base, such as a data base containing stock quotations. The caller may signal the system to access one or more quotations from the data base whereafter the system converts the quotation to an audio voice signal which is output to the caller's phone line.

As can be appreciated, for such systems the interaction between a caller and the system may become quite complex. As a relatively simple example, if the caller desires to learn if any voice messages are stored for the caller the system may respond with a voice signal such as "you have three new voice messages". In generating this response the number "three" is a variable which is determinable at the time that the caller is connected to the system.

Furthermore, the word "messages" is also a variable in that if only one voice message is pending the singular form "message" should be returned and not the plural form. It can thus be appreciated that the ability to accurately define a series of system responses to an incoming call is an important aspect of such a voice response system.

Also, it is preferable that a voice applications writer be able to create and modify the system responses in a relatively uncomplicated and time efficient manner. That is, the operator of the system should be able to interact with the voice response system to create and modify voice responses in a manner which does not require the direct assistance of the provider of the system or the direct assistance of skilled programming personnel.

Many business applications can be automated with voice processing technology. A business can use voice processing equipment to call its clients and deliver or solicit information. Alternatively, business customers can call into a firm's voice processing unit to obtain information, place orders, or transfer to human service agents or other response equipment. Other applications can employ voice processing equipment to exchange information with other call handling equipment without human intervention.

In most cases, the call originating or call transferring automated equipment must be able to communicate information to a user on the basis of dynamic information entered by a user. An example of a prior art call processing system that can benefit from incorporating the subject invention is US Patent 4,627,001, which discloses a voice editing data system using a display system for editing recorded speech. US Patent 4,779,209 discloses another system for editing voice data. US Patent 4,853,952 discloses yet one more text editing system for editing recorded voice signals. US Patent 4,766,604 discloses yet another voice message handling system which includes voice prompts. Finally, US Patent 4,920,558 describes a speech file downloading system wherein static voice prompts are recorded.

In a voice-based computer application system it is often necessary to edit recorded voice prompts in order to produce natural-sounding results. However, this voice data is typically stored in compressed form in order to minimize data traffic and storage consumption within the system. Moreover, repetitive editing of the same voice information introduces increasing distortion into the edited result owing to the approximations in the compression and decompression techniques involved. These distortions may be particularly severe in high-rate compression.

The present invention seeks to address these problems and accordingly provides, in one aspect, a method for editing compressed voice information, comprising the steps of: selecting a segment of compressed voice information; decompressing the compressed voice segment and displaying the decompressed voice information on a display; marking a portion of the displayed voice information; calculating the location and extent of a portion of the compressed voice segment corresponding to said marked portion; editing the marked portion of the decompressed voice information; and correlating the editing actions from the decompressed voice information to the compressed voice information.

In a second aspect of the invention, there is provided an apparatus for editing compressed voice information, comprising: means for selecting a segment of compressed voice information; means for decompressing the compressed voice segment and displaying the decompressed voice information on a display; means for marking a portion of the displayed voice information; means for calculating the location and extent of a portion

of the compressed voice segment corresponding to said marked portion; means for editing the marked portion of the decompressed voice information; and means for correlating the editing actions from the decompressed voice information to the compressed voice information.

Using the Voice Application Generator tool of the preferred embodiment of the invention, the user (i.e., the person performing the voice editing) invokes a voice generation screen. She selects a voice segment of interest, and the "modify" option. A window is opened and the analog wave form of the selected voice segment is presented. This wave form is created by causing the selected compressed voice segment to be looped through voice decompression hardware and returned in clear channel (i.e., uncompressed) form. Both the compressed and decompressed voice forms are retained in memory.

The user proceeds to edit the displayed voice wave form by marking the portion of interest on the screen. Tools are available (e.g., ZOOM) to improve the user's ability to identify the exact position to be marked. The marked segment may then be deleted, for example, or copied into another segment in second voice editing window. In either case, pointers are established at the selected marker positions of the displayed voice segment and in the corresponding positions of uncompressed voice segments.

The voice data is treated as a stream of fixed-length micro-segments, where there is a predictable correlation between the positions of the compressed and uncompressed data. In the implementation at hand, these micro-segments are 20 ms. in length. Editing is accomplished by modifying micro-segments in both the compressed and uncompressed segments simultaneously.

When the user is satisfied with the result, the edited wave form is redrawn on the screen. The user may then SAVE, the result, and the entire segment is rewritten to the data base, replacing the previous version. Only the compressed version is written, thus eliminating the need for a subsequent pass through the compression hardware with the associated compounding of distortion.

A preferred embodiment of the invention will now be described, by way of example only, with reference to the accompanying drawings in which:

Figure 1 is a system diagram of the voice application enabler apparatus in accordance with the subject invention;

Figure 2 is a system diagram of the internal components of the voice application generator in accordance with the subject invention;

Figure 3 is a system diagram of the internal components of the voice application enabler in accordance with the subject invention;

Figure 4 is a block diagram of a state table in accordance with the subject invention;

Figure 5 is a block diagram of the data base files in accordance with the subject invention;

Figure 6 is a system diagram of the general purpose server development process in accordance with the subject invention;

Figure 7 is an illustration of a prompt segment editor display in accordance with the subject invention;

Figure 8 is an illustration of a prompt editor panel in accordance with the subject invention;

Figure 9 is an illustration of a state editor panel in accordance with the subject invention;

Figure 10 is a block diagram of a phone service application in accordance with the subject invention;

Figure 11 is an internal state table in accordance with the subject invention;

Figure 12 is a block diagram illustration of a how a complex variable is played in accordance with the subject invention;

Figure 13 is a block diagram of the national language support parameter numbers in accordance with the subject invention;

Figure 14 is a block diagram of the state table managers in accordance with the subject invention;

Figure 15 is a flowchart of a play variable in accordance with the subject invention;

Figure 16 is a flowchart of editing voice segments in accordance with the subject invention;

Figure 17 is a flowchart of national language support setup in accordance with the subject invention;

Figure 18 is a flowchart of national language support application development in accordance with the subject invention;

Figure 19 is a flowchart of national language support execution in accordance with the subject invention;

Figure 20 is a flowchart of preparation for playing complex variables in accordance with the subject invention; and

Figure 21 is a flowchart of a playing complex variables in accordance with the subject invention.

Referring to Figure 1, the apparatus for performing a voice related application includes an RISC System/6000 (trademark of IBM Corporation) 10 with on board RAM for executing a variety of applications including AIXwindows (AIXwindows is a trademark of IBM Corporation) 60, Voice Applications Enabler (VAE) 70, and a data base 50 for storing and transferring static and dynamic voice information to the memory of the RISC System/6000. The data base 50 also contains means for storing rules for vocalizing the voice information. The rules

are interpreted by the Voice Applications Generator (VAG) 80 and used to vocalize the dynamic and static voice messages into the proper user prompt. A terminal 20 is attached to the RISC System/6000.

Using standard IBM hardware and operating systems, the Voice Application Enabler (VAE) connects callers with application servers and manages the sessions, with the telephone as the interactive medium. Principal elements of the system are the Application Server Interface (ASI), Voice Application Generator (VAG), and the Host, which is a host system providing data base access and storage.

The VAE provides facilities and connectivity for customer implementation of voice-data applications. It uses telephone lines to connect a customer-premises or central office switch to provide dual-tone multi-frequency (DTMF) signalling recognition for state-of-the-art voice compression and decompression. Customers can develop application scripts and voice prompts with an easy-to-use, high level application-specific language or graphic interface and other development tools.

Several publications describe the operating system tools used to implement the invention. IBM Advanced Interactive Executive for Personal System/2 (AIX PS/2) IBM AIX (AIX is a trademark of IBM Corporation) Operating System Commands, IBM Publication Number - SC23-2025-0. IBM AIX Operating System; Technical Reference -SC23-2032-0 IBM AIX Operating System; Tools and Interfaces - SC23-2029-0.

The Voice Application Enabler (VAE), Prototype, is designed to support interactive transaction processing and to integrate voice and data applications. The major characteristics of the VAE are:

- Telephone switch interfaces;
- Advanced voice compression;
- Telephone used as interactive medium; and
- Application level customer programmability.

The voice-data application addresses two identifiable environments:

- Central office (CO) telephone company enhanced service provider; and
- Customer premises computer assisted telephony, using a PBX or CO switch.

The VAE provides facilities and connectivity for customer implementation of voice-data applications. It uses telephone lines to connect to a switch, and provides dual-tone multi-frequency (DTMF), switch signalling recognition, and voice compression and decompression. In addition, it supports rotary dial telephones when voice recognition is operational. Users can develop application scripts and voice prompts with an easy-to-use, high-level, application-specific language or graphics interface, and other development tools. Users can also conduct application sessions with a Host computer using the telephone. Examples of such applications include voice messaging, voice information services, and data base applications.

In the CO environment, generality of function, scalability, reliability and support of multiple switches are critical requirements, while in the customer premises environment, entry cost, broad functionality and multi-switch compatibility are key elements. Both environments share common technological dependencies: multi-channel operations, real-time response, state-of-the-art voice compression, and telephone channel signalling recognition and control.

The VAE accommodates a variety of line interface protocols ranging from a few analog voice channels to multiple T1's and Integrated Services Data Network, both domestic and worldwide. It is easily tailored by the customer to suit his or her application.

The VAE is designed as a multi-lingual system. Support is provided for single-byte, left-to-right languages only. Special provisions will be made for other language requirements, such as bidirectional Hebrew and Arabic, or other single- or double- byte languages. The VAE implements enabling language function and translation for hardware and software and for translation of customer and service information, such as messages, help panels, documentation and nomenclature.

Typical Scenario

In a typical scenario, a user calls a telephone number and is directed by the CO switch into the VAE system. The voice system interacts with the user through dynamically assembled voice prompts and phrases, and the user interacts with the voice system through the voice and the DTMF touch pad on his or her telephone.

VAE allows the caller to interact with existing customer data base applications, and to transfer to a live operator when needed. For example, in a voice-messaging application, the user may be prompted to leave or retrieve a message. In a voice response application, he may hear a weather forecast or inquire about his bank balance or order a pizza.

In each case, sub-second response and unbroken voice streams allow users to interact with the system as a single image without regard to call routing. In this way, each customer is able to program the system so that it is user friendly, and interacts with him in a manner appropriate to the application being performed.

T1/CEPT Signalling

VAE is required to interface with a variety of CO switches. The preferred medium is T1-D4 in the United States and CEPT in Europe.

Telephony Interface Protocol

Line information comes from a separate Simplified Message Service Interface (SMSI) channel in each implementation. This signalling also comes from the call setup information from existing tariffed signalling services, such as the SMSI.

As Integrated Services Data Network (ISDN) evolves into general use, the VAE should adopt its interface protocol as the preferred technique. Some CO switches may not support the required interface protocol services in conjunction with the digital T1 service. In such cases, personalized greetings and message waiting indications are not likely to be implemented. To accommodate analog telephone systems a channel bank is required to convert analog signals to digital signals.

Topology and Scalability

The VAE supports voice channels ranging from a few voice channels to over a thousand. Where redundancy and capacity requirements are low, the system is designed to operate on a single, standalone hardware platform. Where they are high, the system design permits $n + 1$ redundancy and distributed function, so that very large configurations retain a single system image to the user.

Design Structure

VAE is an application system, and it operates entirely under AIX, Version 3. VAE operates on IBM RISC Model (RISC System/6000). Users interact with the VAE using a high-level application language consisting of pre-programmed primitives. In addition, the customer will define and program application servers to supplement those provided in the standard release.

Each VAE node will support up to 72 voice channels in the United States and 90 voice channels in Europe. Overall system performance is application dependent. Limited multi-lingual support is intrinsic to the design and will be in place for subsequent releases.

SYSTEM ARCHITECTURE

The VAE system architecture is shown in Figure 1. The system consists of three logical elements:

- O Front-end ASI 90;
- O Back-end Host data base server 50; and
- O Operations Console (VAG 80, SAF, UAF, and OAM).

The most general manifestation of the system architecture allows for multiple ASI's and Host servers. In the general architecture, the ASI and Host server components are linked using one or more local area networks or by direct attachment to a data base Host.

The hardware base for the subject invention is the IBM RISC System/6000 10. The operating system is RISC/6000/AIX, a UNIX derivative, with significant real time multi-tasking capabilities. The invention employs a back-end data base server 50 designed to exploit a set of sequential files, which are accessed at the field level, and indexed sequential files, which are accessed at the field and/or record level. Field-level access provides relational data base management system capabilities. The function of the data base server is to retrieve and store application scripts, system parameters, voice data, and subscriber information.

The database server 50 provides intelligent data base services to the ASI and VAG on demand. The data base server function operates on the same physical hardware platform as the ASI application. An architected interface between the ASI functions and the data base server functions is maintained in common for combined and distributed configurations. The data base functions are provided by IBM. A custom server architecture enables customers to define and develop functions of their own design using the same Data Base Management Services as used by the VAE system servers. The peculiarities of the local server implementation, however, are insulated from the ASI by the architected interface.

FUNCTIONAL CHARACTERISTICS

The basic facilities of the VAE system, as illustrated in Figure 2, provide support to end-user applications including voice information services, voice messaging and voice interactive transaction processing, similar to IBM's Voice Response Unit (VRU). Other operations are built on these basic functions or their variants.

Call Transactions

Call transactions begin with the establishment of an active telephone connection with the CO at the ASI. Identifying call information, such as called party number, determines the selection of a script that describes and controls the specific type of transaction. An ASI is able to handle all types of transactions for all subscribers and callers. There is no prior binding of caller or transaction type to specific servers or to trunks/channels.

The call transaction control function in the ASI follows the script for the duration of the active transaction. The script contains the list of actions to be performed, such as play a prompt or receive digit, and the conditional sequencing of the actions in the list. The action library resides in the ASI and the script, being customizable for each transaction and subscriber, must be fetched from the data base through the Host server. Common prompts are cached in the ASI also, while customized prompts and greetings reside in the data base.

Compression and Decompression

Compression and decompression of voice signals occur within the ASI immediately after receipt from the CO and before being sent back to the CO. Multiplexing and de-multiplexing of the T1/CEPT channel are also done at this point. Thus, the CO always perceives normal voice. DTMF detection is incorporated in the line interface hardware, permitting user interaction with the system.

Digit information is removed from the information by the control function. In the case of messaging, the compressed incoming message is stored in the data base in segments during the recording. Recorded messages are retrieved in segments, each decompressed and sent out while successive segments are fetched in a manner ensuring smooth voice regeneration. Prompts, greetings and other voice responses are played back in the same manner.

The compressed voice segments flow over the link between the ASI and Host server. In the Prototype, the link between the ASI and the Host is a logical construct designed to maintain compatibility and configuration flexibility in later releases. Functions in the Host provide data base access and other support needed by the ASI applications, such as updating user profile information. In addition to the voice segments, control information flows between the ASI and Host functions during the transaction.

Application support from the Host may schedule subsequent activity to complete the transaction, such as interpreting and passing the transaction to a remote data base system. The interface provided in the General Purpose Server allows the customer application to interact with the VAE. It receives transaction information from the ASI, activates functions imbedded in the application program, and then returns the results back to the ASI.

MAJOR COMPONENTS

Figure 3 illustrates the major components of the VAE system:

- Application Server Interface **200**;
- Host **210**;
- Operations, Administration, and Maintenance (OAM) **220**;
- System Administrator Facilities (SAF) **230**;
- User Administration Facilities (UAF) **240**; and
- Voice Application Generator (VAG) **250**.

This section introduces each of these components with a brief description. Following the VAG section is a section listing and describing the VAG application actions.

APPLICATION SERVER INTERFACE (ASI)

The telephony system connects to the VAE through the Application Server Interface (ASI) **200** of Figure 3. These connections are the trunk channels over which telephone calls are transmitted. In addition, Simplified Message Service Interface (SMSI) **205** is a separate signalling link over the serial I/O port that conveys information relative to the telephone calls, such as source and destination identification. The ASI **200** handles mul-

multiple calls at one time, performs all the logic, and contains all the states for each call. The number of message servers configured in an ASI system is a function of the number of connected trunks. Each ASI can connect to a minimum of one T1 carrier with twenty-four channels in the United States and to CEPT with thirty channels in Europe.

5 Compression and decompression of voice traffic to and from the trunk channels also occurs in the ASI 200. The ASI does not provide disk storage for the voice messages. Voice messages are sent to the Host for storage and retrieved from the Host 210 when requested. The voice messages are always conveyed between the ASI and the Host 210 in a compressed form, with a compression ratio of five to one plus pause compression. Clear channel mode(1) is included in the VAE architecture.

10 ASI is built on a RISC/6000 base and uses standard IBM components, with the addition of required cards for switch connections and voice compression. The ASI consists of:

- O Voice Card Set 260,
- O Voice Driver 270,
- O Signalling Driver 280,
- 15 O Cache Manager 300,
- O Node Manager 310,
- O State Table Manager 320,
- O Prompt Directory Manager 330, and
- O Data Base Interface Manager 340.

20 Voice Card Set

The Voice Card Set performs voice compression/decompression and signalling management on multiplexed telephone channels. It consists of the following cards:

25 VOICE SERVER CARD (VSC):

Performs voice signal processing and interfaces to the ASI RISC System/6000 Host. It contains a master signal processor (SPM, or SPO) and five slave signal processors (SPS, or SP1-SP5). Three slave processors are located on a daughter card.

TRUNK INTERFACE CARD (TIC):

35 This is the VSC daughter card that performs the interface between the VSC and the multiplexed digital telephone line. There is a version for T1 and a version for CEPT. The TIC features an Intel 8751 microcontroller.

VOICE SERVER CARD ADAPTER (VSCA):

40 Plugs into the microchannel bus and acts as an interface between the Host 210 and the VSC 260. The VSCA has no processor, and it can be thought of as a translation mechanism between the microchannel bus and the VSC bus 260. The VSC 260 and TIC combination is referred to as the VPACK. The VPACKs (up to six) reside in a 7866 rack-mount modem package.

Voice Driver

45 The Voice Driver code is written as an AIX device driver. It operates synchronously with the voice card set, fielding interrupts and moving voice data and status information to and from the voice card set. The interrupt routines perform the following functions:

- O Read the status of each channel;
- 50 O Move raw voice data to and from the appropriate channels;
- O Move encoded voice data to and from appropriate channels;
- O Retrieve voice signals from cache for playing cached prompts;
- O Receive and send signalling information for each voice channel;
- O Process signalling information for each voice channel;
- 55 O Read and write VPACK card level status and commands; and
- O Field and post alarm conditions.

Signalling Driver

The Signalling Driver software 280 is the code that interfaces with the SMSI 205 in order to process messages to and from the Central Office (CO). SMSI is the telephone company's protocol for messages that travel to and from the CO.

There are four types of messages that travel to and from the CO. They are:

CALL HISTORY:

- Travels from the CO to the ASI. Information that identifies a call as it comes into the ASI 200 from a CO on a T1 line.

MESSAGE WAITING INDICATOR (MWI):

- Travels from the ASI to the CO. It signals the CO to light the message waiting lamp, initiate a stutter tone on the customer's telephone, or turn off the message waiting lamp.

NEGATIVE MWI:

- Is the response from the CO to the ASI that the CO is unable to process the corresponding MWI message.

UPDATE SWITCH:

- Travels from the CO to the ASI. It requests the transmission of all pending MWI messages from the ASI.

Channel Processes

- Channel processes are the vehicles for the ASI's application logic with one active Channel Process for each active session. The Session Manager is the primary execution component in the Channel Process and is responsible for interpreting the user's application script into a script table. The application script, which is prepared using the VAG, consists of actions, parameters, and a table of conditional program flow parameters for the various possible return codes, or edges.

- Examples of edges include: key x pressed, caller hung up, and time out. Code for the State Machine and all the actions is re-entrant and is shared among all Channel Processes. A portion of a typical state table is shown in Figure 4 for the bus schedule.

INTERNAL STATE TABLE:

- The first state table (internal state table) executed is hardcoded in the State Machine. It contains the following actions:

IDLE	Wait for notification from the Node Mgr.
ANSWERCALL	Get the user profile, get the state table, then answer the call.
PLACECALL	Get the user profile, get the state table and place the call.
ENDCALL	Ensure the line is on the hook, reassign the line to the Node Manager, and do any other clean-up work that is needed.

- At Channel Process start-up, the State Machine is called with a pointer to the hardcoded internal state table and the first entry executed is the Idle action. When notified, Idle wakes up, checks if the request is to answer or place a call, and then returns the appropriate return code to pass control to the AnswerCall action for incoming calls. All the actions that follow depend on the return codes. These return codes indicate that an action is complete or that some external event has occurred.

When the entire transaction is complete, the CloseSession action is executed. This is the last action that is performed in a state table, at which time, control is returned to the State Machine's internal state table. The State Machine then executes the EndCall action and then, again, the Idle action.

- The State Machine also supports nested state tables. By defining state tables, the user can develop a library of commonly used functions and link them together to create larger and more sophisticated applications. This is accomplished by making the State Machine recursive and by using the CallStateTable and ExitStateTable actions.

Cache Manager:

The Cache Manager controls the storage and mapping of memory for voice segments. Voice segments are stored in shared memory. The amount of cache required to store voice segments is determined by the customer's application requirements. The system configuration facility permits the adjustment of the amount of cache.

Caching is transparent. The requesting function is presented with a pointer that indicates a list of voice segment pointers. If the voice segment is already in memory, the return is immediate. If the voice segment is not in memory, the first segment is retrieved from the Host data base and its pointer is passed to the requesting function. The Cache Manager continues to read in the rest of the voice segment independent of the requesting function.

The Cache Manager stores voice data in 4K buffers and contains two control blocks. The first control block is a directory that contains a series of maps that describe the location of each of the voice segments in memory. The second is a short-term control block that is used to make a logical connection between the segments that are requested and their requestor(s).

An adjunct process called compress voice segments run at predetermined intervals to reorganize the cache memory in order to recover memory space that has been unavailable because of fragmentation.

Node Manager:

The Node Manager is considered the parent of the other processes running in the ASI, including the Channel Processes. The Node Manager is responsible for loading and initializing the entire system; reading and interpreting initialization parameters from the system tables; and reading permanent voice segments, state tables, and prompt directories into memory. It assigns Channel Processes to events when activity is indicated on an idle voice channel and serves as the catcher for unsolicited requests that come from the Host. Exceptional conditions, such as alarms and alerts, are processed first by the Node Manager. The Node Manager also sets a deactivated status on selected channels for maintenance.

The buffer pool, which is managed by the Node Manager, is a pool of 4K buffers available for allocation to other system managers and Channel Processes. Any process, such as a Channel Process, or any system manager requests a buffer through a function call. Then, when the process or manager no longer needs its 4K buffer, it returns it with another function call.

State Table Manager:

The State Table Manager provides access to state tables and maintains copies of as many state tables as needed, up to a predefined limit. The Channel Process accesses state tables to interact with calling parties. If a Channel Process requests a state table that is not currently in memory, the State Table Manager requests that state table from the data base and notifies the Channel Process when it is available.

The State Table Manager continues to read in state tables until the predefined number is in memory. When this number is reached, the State Table Manager replaces the non-active tables with new tables. If there are additional requests for state tables that are not in memory, the State Table Manager removes the least recently used state table, if not currently in use, and requests the new one to be read. The Channel Process is responsible for notifying the State Table Manager when it is no longer using a state table.

Other features of the State Table Manager include processing an invalidate state table request, which means that a state table has been updated by the VAG, and differentiating between tables that are fixed in memory from those that are not.

Prompt Directory Manager:

The Prompt Directory Manager provides access to prompt directories and variable segment directories, and maintains copies of as many prompt directories and variable segment directories as are needed, up to a predefined limit. The Channel Process accesses prompt directories and variable segment directories to play prompts for calling parties.

If a Channel Process requests a prompt directory and a variable directory that are not currently in memory, the Prompt Directory Manager requests them from the data base and notifies the Channel Process when they are in place. The Prompt Directory Manager continues to read in prompt directories and variable segment directories until the predefined number is in memory. If there are additional requests for prompt directories and/or variable segment directories not in memory, the Prompt Directory Manager removes the least recently used

ones, if not currently in use, and requests the new ones to be read.

The Channel Process is responsible for notifying the Prompt Directory Manager when it is no longer using a prompt directory and a variable segment directory. Whenever a prompt directory is updated, the Prompt Directory Manager is notified. If the prompt directory is in memory, the Prompt Directory Manager flags it as invalid-
 5 dated. A subsequent request from a channel for this prompt directory causes the Prompt Directory Manager to request the updated prompt directory from the data base.

Prompt directories are kept in memory in 4K buffers. The number of 4K buffers required for a prompt directory depends on the number of prompts in the prompt directory and the length of each prompt. There is no limit to the number of prompts a prompt directory may have. The number of 4K buffers a prompt directory can occupy
 10 is limited by the number of 4K buffers available.

Prompt directories are either permanent or temporary. Once read into memory, permanent prompt directories stay fixed in memory. Temporary prompt directories remain in memory until the memory they occupy is needed to satisfy other prompt directory requests or the Prompt Directory Manager is notified by the Node Manager that it must release 4K buffers. In either case, only temporary prompt directories that are not currently in
 15 use are released.

Data Base Interface Manager

The Data Base Interface Manager provides the interface between the ASI processes and the data base servers. The common protocol for processing all requests for data base service and the responses to these
 20 requests is the Data Processing Request Block (DPRB).

DPRB Structure

The Data Base Interface Manager maintains a control table to keep track of outstanding requests that require response from the data base servers. The information contained in this table includes:

- O Requestor id
- O DPRB number; and
- O Time of the request.

The requestor id enables the system to return the request once the Data Base Interface Manager gets a response from the data base servers. Requests and responses may consist of a single DPRB or multiple DPRB's. The time of the request allows the system to pinpoint when a request has timed out. There are some requests that do not require a response from the data base servers. These requests are passed on to the servers without recording them in the control table.

The amount of memory required by the Data Base Interface Manager for the control table is a function of the maximum number of requests that require a response and the maximum length of time the Data Base Interface Manager is required to wait for a response before declaring a time out and returning the request to the requestor.

HOST SYSTEM

The Host system consists of the following subcomponents:

- O Data Base Management System
- O Message Router
- O VAE Data Base Servers
- O Custom Servers

Data Base Management System

The Data Base Management System consists of the Data Base Service Manager function and the indexed sequential and sequential file structures. The Data Base Service Manager, or server, receives DPRB's from, and sends DPRB's to the Data Base Interface Manager. Data base servers provide various data base services to the ASI and the VAG. These services include retrieving, creating, deleting, and updating data in a file, as well as performing various data base backup and recovery functions.

The Data Base Service Manager and the indexed sequential and sequential files access methods provide data access functions. Figure 5 illustrates an overview of the data base files and their relationship to one another.

Overview of the Data Base Files

A file can be accessed for services at the file level, the record level, or the field level, depending on what access method is used to implement the file. For example, the indexed sequential access method is used primarily to access a file at the record level. If there is no need to access a file at the record level, the sequential access method is used. User profile files and mailbox files are accessed at the field level.

Message Router

The Message Router is implemented as an AIX queue where messages are forwarded to the AIX queue by the Data Base Interface Manager. The messages, which are the service requests from either the ASI or the VAG, are formatted in either the direct or indirect form. The direct (or long) form is where the input parameters are contained within the message itself. The indirect (or short) form is where the input parameters are contained within the DPRB.

In either case, the data base server searches the queue for its request based on the message type and receives it into its task space for processing.

Data Base Servers

There are four data base servers in the VAE system. They are:

- O State Table and Prompt Directory Server;
- O User Profile/Mailbox Server;
- O SGAM Server; and
- O VAG Server.

STATE TABLE AND PROMPT DIRECTORY SERVER:

The State Table and Prompt Directory Server retrieves a state table, a prompt directory, or a variable segment mapping file from the data base and returns it to the ASI through the Data Base Interface Manager. The server uses the sequential file access method to accomplish this task.

USER PROFILE/MAILBOX SERVER:

The User Profile Server retrieves, updates, or deletes a record of a user profile file for the ASI through the Data Base Interface Manager. A record within a user profile file can be updated at the field level. For all functions, with the exception of retrieval, the ASI can request the server to return an acknowledgment of completion. The server uses the unique user id to search for a record in the user profile file.

The Mailbox Server contains the mailbox data base that provides the link between the user profile files and the message files. The mailbox data base is a collection of mailbox entries, or message headers that describe each message that arrives at the VAE. Pertinent information that is contained in the message header includes the message id, the sender's id, the receiver's id, and the message status code.

SGAM SERVER:

The Segment/Greeting/Audio Name/Message (SGAM) Server retrieves, creates, updates, deletes, queries, renames, or copies a data unit for ASI through the Data Base Interface Manager. A data unit can be a voice segment, a greeting, an audio name, or a message. Each voice segment, greeting, audio name, and message contains a set of voice records, and can be searched by a unique key. For example, the unique key for a voice segment is composed of a segment id and a sequence number.

With the exception of retrieval and query, the ASI requests the SGAM Server to return an acknowledgment of completion. The server uses the indexed sequential file access method to search for a data unit. Voice segments are stored in different files according to language code and compression ratio, greetings and audio names according to compression ratio, and messages according to recording date and compression ratio.

VAG SERVER:

The VAG Server retrieves, creates, updates, or deletes either a record in a file or a file. An example of a file might be a state table or a prompt directory. An example of a record of a file might be a voice segment in

a voice file or record in a user profile. For all functions, with the exception of retrieval, the VAG requests the server to return an acknowledgment of completion. The server primarily uses the sequential access method to access files and the indexed sequential access method to access records in files.

5 Custom Servers:

In addition to the standard VAE data base servers, the following two special servers allow customers to connect their own applications to the VAE system:

- O 3270 Server
- 10 O General Purpose Server

3270 SERVER:

The IBM 3270 Server is a turn-key method of interfacing the VAE with existing Host-based applications that use 3270-type displays. Other methods require the customer to develop custom interface routines. The VAE uses the AIX Host Connection Program (HCON) to support the functions of the 3270 Server. Communication with the Host is provided by an AIX systems network architecture (SNA) service's token ring and synchronous data link control (SDLC) links through the RISC/6000 token-ring and multi-protocol adapter.

Up to twenty-six concurrent sessions are supported by token ring or SDLC links. Any 3270 terminal or controller supported by AIX HCON may be used by the VAE. To create an application, the customer uses the VAG to customize the ASI interaction with the 3270 server. To accomplish this, the customer uses VAG functions such as Host session parameters, logons, expected Host sequences, error handling, and data field locations and types. Neither programming expertise nor the use of a compiler is necessary to create VAE applications using the 3270 Server.

25 GENERAL PURPOSE SERVER (GPS):

The General Purpose Server (GPS) is a construct designed to provide open-ended access to the facilities of the VAE. It is used to support local and remote data base access through the use of custom servers. In general, the customer is responsible for creating the custom servers needed using the Custom Server User Interface in conjunction with his/her own programming logic and the VAE Application Generator.

The resulting custom server operates within the Host or pseudo-Host component of VAE. Server logic is provided in the form of source or object modules specified by the customer using a compatible programming language processor, such as: COBOL, FORTRAN, C, or PASCAL. Any operating system service or facility may be used, with the condition, that the design must conform to the performance requirements of the application.

Interaction with the VAE telephony environment is provided automatically using the pre-processor phase of server build. In a server development session, the customer's logic modules are imported, the Host interaction is specified with the front-end script, and the combined specification is submitted to the build process.

The GPS design can be explained by dividing it into two parts: the GPS components and the GPS architecture. This section provides a brief description of the GPS components followed by an overview of the GPS architecture. The GPS components are:

O APPLICATION PROGRAM INTERFACE (API)

This interface module is part of the VAE system and is stored in a library in the VAE system data base.

O USER APPLICATION MODULES (UAM)

These interface modules are a collection of routines supplied by the customer to provide access to the existing customer application system.

O GPS PREPROCESSOR

This is an Application Specification File parser composed of AWK programming files.

O APPLICATION SPECIFICATION FILE (ASF)

This file defines and specifies the parameters and information necessary to generate the custom server. It is created by the customer using the Custom Server User Interface.

O APPLICATION FUNCTION/SUBFUNCTION FILES (ASF), (AFF)

These files contain the function id for the customer application system and the information for invoking the User Application Modules. When creating a custom server, the customer is responsible for providing the User Application Modules and creating the Application Specification File. The GPS provides the remaining components and all the processing necessary to generate and implement the custom server.

The GPS design architecture is illustrated in Figure 6. The GPS is divided into four stages:

- O Custom server development;

- O Script processing;
- O VAE system initialization; and
- O Runtime facilities.

Custom server development consists of creating the Application Specification File (ASF) and storing it in the VAE system data base. This is implemented using the Custom Server User Interface. When the Application Specification File is created, it is subjected to a BUILD process.

During the BUILD process, the GPS Preprocessor reads the Application Specification File (ASF) and generates the Application Function/Subfunction Files and the Application Control Program. The C compiler compiles the Application Control Program and a MAKE utility links it with the User Application Modules to build the executable file as the custom server.

The script processing consists of generating a script with the action parameters necessary to link the application with the custom server. The main action parameters are the SendData and ReceiveData actions. Also, when generating the script, the script reads the Application Function/Subfunction Files and uses the Custom server (bus scheduling) and subfunction name (get-city, get-schedule) to generate the script.

The Application Function/Subfunction File directs the script programmer to give input parameter(s) as needed by the subfunctions. At VAE initialization, the system sets up the system parameters and allocates the resources necessary to implement the custom server. Finally, at runtime, the application runs the script table, sends the RB (request block) that contains the custom server number (function id), the transaction number (subfunction id), and the parameter data to the custom server. The custom server then parses the parameter data to the input parameters and passes the input parameters to the customer application system.

When the customer application is complete, the custom server passes the RB header and the return parameter back to the VAE application.

OPERATIONS, ADMINISTRATION AND MAINTENANCE

Operations, Administration and Maintenance (OAM) functions for the VAE system includes configuration management, performance management, and error management. OAM performs the following functions:

- O Provide a console interface for system operation and command execution;
- O Statistics collection;
- O Statistics reporting; and
- O Error management.

Console Interface

The OAM Console Interface provides the environment for the system administrator to continuously monitor the status of the VAE system. It also allows the system administrator to take appropriate action in response to alerts and warnings. The status information is displayed graphically and refreshed at regular intervals. Status information includes:

- O Percentage of buffer pool available;
- O Percentage of disk space in use;
- O Number of active lines on each VSC and the status of each line;
- O System performance statistics such as voice segment cache memory requirements; and
- O System configuration information.

The OAM Console Interface operates in the AIXwindows environment. The screen controls and options are grouped by function and purpose and are activated by input from either the keyboard or the mouse. The main screen is divided into three areas:

- O Menu bar;
- O System status display area; and
- O User selected status display area.

The system administrator may execute administrative commands to change the configuration and/or operational characteristics of the system. Administrative commands include:

- O Start, stop, or block a channel;
- O Start or shut down the system;
- O Control system resource usage (request buffer release);
- O Request reports;
- O Change system parameters;
- O Add, modify, and delete user profiles and mailboxes;
- O Define classes of service; and

- O Direct the console display to other devices.

Statistics Collection

5 Collection of statistics concerning system operation and resource utilization is accomplished by a process that executes on a periodic basis. This process reads shared memory locations and interfaces with other processes; such as, the Node Manager and the State Table Manager.

Data accessed by this process include the:

- O Buffer pool;
- 10 O Disk space;
- O Voice segment cache completion percentage;
- O State table hit/miss ratio;
- O Prompt directory hit/miss ratio; and
- 15 O Trunk error performance.

Statistics Reporting

In addition to the error logging process described earlier, OAM records events in an event log. The event log includes call completion records, console operations events, and threshold violations.

Error Management

25 The VAE error management system fields all detected errors in the system. Each hardware and software component is designed to identify error conditions. For example, the VSC continually monitors the trunk status and presents error conditions (in the form of alarms) to the VAE software. Similarly, each VAF software component tests for invalid inputs, system-related failures, illegal requests, or resource availability problems. While the architecture allows VAE to present err_enroll as an interface to the custom server writer, the Prototype does not allow the customer to use this error recovery service.

30 VAE SOFTWARE ERROR RECOVERY

Coverage of VAE software identified failure conditions is targeted at 100 percent. That is, all identified error conditions are detected and recovery actions is assigned. All recovery actions can be grouped into the following five general types of recovery procedures:

- 35 O Logging;
- O Process local recovery;
- O Multiple process recovery;
- O Single process restart; and
- O System restart.

40 In the event that a software module receives invalid input, the VAE may log the problem, disregard the transaction, and notify the requestor of the error condition. An intermittent failure, such as a failed data base query, may initiate a process local recovery procedure, in which case, the requesting process may retry before escalating the problem to OAM.

45 A shortage of shared buffers in the buffer pool may require a multiple process recovery procedure. In this event the requesting process notifies a system management process, such as the Node Manager, which, in turn, requests other processes to free unused buffers to make them available to the original requestor.

50 Recovery for VAE software failures caused by data corruption, logic errors, or exceeding designed boundaries is to quiesce the Channel Process, if possible, and re-initialize it. This is an example of single process restart. The same is true of failures caused by event overflows or a missed interrupt. The error management system can generally restart any failed non-system management VAE process. A failure in system management processes requires a full system restart.

AIX AND SYSTEM HARDWARE ERROR RECOVERY:

55 Coverage of non VAE problems is limited to those errors detected by AIX and the hardware. AIX and system hardware errors may or may not be recoverable. Disk and controller failures, memory checks, bus checks, and other hard errors are almost always fatal. Fatal errors require manual intervention. When intermittent errors occur during normal software execution, the software may retry at least once before escalating the problem to

OAM.

Failures caused by insufficient resources may or may not be fatal. Recoverable conditions, such as a disk full conditions causes all attempts to record voice or update user profiles to fail. Under such conditions, and when more 3270 emulation sessions are required than are currently enabled, a system busy announcement may be played, if specified by the customer script, and input is declined until resources are available.

Error conditions that occur at the telephone line trunk interface (VSC) is analyzed and a determination of Red Alarm, Yellow Alarm, or Alarm Indication Signal is made. The occurrence of a Red Alarm causes the VAE system to disconnect all calls on the unit and make all channels busy at the originating end for the duration of the condition. The VAE system automatically restores service after the alarm has been cleared.

Certain conditions can trigger a flood of error that can overwhelm the logging device (or Host). Provisions are made to set thresholds for error reporting. These threshold settings are variable and can be reset. The console operator is notified of the state of all channels and about all errors. The operator then diagnose the error and initiate the appropriate error recovery action.

15 SYSTEM ADMINISTRATOR FACILITIES (SAF)

The SAF is a VAE application that allows the system administrator to establish, maintain, and support the following system functions:

- O Application Profiles;
- O Administrator Profiles;
- O National Language Support Setup; and
- O System Configuration.

The SAF is menu-driven and is essentially a graphics workstation application using AIXwindows as the basis for its design. Access to the SAF is from the main default window for the VAE applications.

25 Application Profiles

An application profile, is used at ring time to set up the actions that are performed after the telephone is answered. The most important information stored in the application profile is the state table id (the application to run) and the entry point in the state table. The application profile is stored in the same file as the user profile. Unlike the user profile, it does not describe the user; instead, it describes the application.

The application profile user interface consists of a panel where the system administrator enters the profile data. This panel consists of a list of the existing application profiles and the actions: ADD, DELETE, MODIFY, and SEARCH. On the right-hand side of the panel is a work area where the application profile information is entered and displayed. The system administrator may select an existing application profile from the list and perform an action on it, or by using the ADD action, She may create a new profile.

- Each application profile includes:
- O Phone number;
 - O Application name;
 - O Comment field;
 - O Language ID;
 - O State table to be used with the application;
 - O Release number for the state table; and
 - O Entry point into the state table.

45 National Language Support (NLS) Setup

The VAE system processes, simultaneously, applications consisting of multiple languages. For this reason, the National Language Support (NLS) setup program is designed and implemented. The NLS setup program duplicates an application in a new language and then allows the system administrator the option of updating the application using the new language. NLS is designed in the same way as the other SAF programs using AIXwindows.

When NLS is selected from the SAF menu, the system administrator is able to modify screens for an existing language or create screens for a new language. There are two parts to the NLS program. The first part allows the system administrator to change the language displayed in the panel fields of the application. Using the NEXT action on the NLS panel, she is presented with each panel field that can be translated. Examples of these fields are MODIFY and DELETE.

The second part of the NLS program is the translation of the application text into the new language. The

system administrator is presented with each panel of the application, where she can change the language of the text for each panel. Examples of the text that is translated are state purpose, action name, and edge name.

In addition to changing the language of the text for an application, the system administrator can also change the playback characteristics of the voice information for the chosen language.

System Configuration

System configuration is a menu option that allows the system administrator to edit configuration parameters. The configuration parameters are grouped by logical component. Default configuration groups are:

- VAE configuration;
- ASI configuration;
- VSC configuration; and
- Language configuration.

The configuration panels for all the groups are the same: on the left-hand side of the screen, appears a list of parameters to select from; and on the right-hand side, there is space to alter the selected parameter.

VAE CONFIGURATION:

VAE configuration consists of system parameters that define the operating characteristics of the VAE system. The VAE configuration program stores and displays these system parameters so that they can be accessed by the system administrator when modifying system configuration.

The VAE configuration program allows the system administrator to configure the system at initial startup and to reconfigure the system when there are modifications. System configuration includes the tasks of choosing the disk space used for recording messages and setting time outs for timed actions. Like the other programs discussed in this section, VAE configuration is a menu-driven program that uses a graphics panel user interface.

ASI CONFIGURATION:

ASI configuration consists of parameters that define the operating characteristics of ASI. The ASI configuration program stores and displays these parameters so that they can be accessed by the system administrator when modifying ASI configuration.

The ASI configuration program allows the system administrator to configure ASI at initial startup and to reconfigure ASI when there are modifications. ASI configuration includes the tasks of specifying the size of the 4K buffer pool, determining the size of cache, and selecting the number of voice cards. Like the other programs discussed in this section, ASI configuration is a menu-driven program that uses a graphics panel user interface.

VSC CONFIGURATION:

The VSC configuration program allows the system administrator to configure the VSC at initial startup and to reconfigure the VSC when there are modifications. VSC configuration includes the task of setting information related to telephone connection, such as the compression type, the number of lines per trunk, and the types of trunks. Like the other programs discussed in this section, VSC configuration is a menu-driven program that uses a graphics panel user interface.

LANGUAGE CONFIGURATION:

Language configuration refers to the way system configuration parameters are defined for each language. English functions as the basic language. Using the language configuration, the system administrator describes how to play variables such as numbers, dates, time, currency, and telephone numbers. For example, the voice format for date includes in which portions of the variable are played (day of week, day of month, month of year, and year), as well as the qualifiers for the variable (day of month as an ordinal or a cardinal number). In addition, a Variable Mapping Table allows the system administrator to define the smallest, or "primitive," elements of the variable (such as months of the year and days of the week).

Utilities Modules

The Utilities Module provides report printing for selected system data files, such as user profiles, application profiles, and system administrator profiles.

USER ADMINISTRATION FACILITIES

The UAF is an interactive program that allows the system administrator to create and maintain the:

- 0 User profile;
- 5 0 Messages sent list; and
- 0 Messages received list.

User Profile

- 10 The user profile function allows the system administrator to create and maintain the data that defines each user. This data is contained in the user profile. Each user profile is uniquely identified by a user id (phone number/extensions) and a mailbox id or id's (a user can have more than one mailbox). Using the user profile function, the system administrator can:

- 0 Search for a user profile in the user profile list;
- 15 0 Select a user profile from the user profile list;
- 0 Specify the contents of the user profile data fields;
- 0 Add a user profile to the user profile list;
- 0 Modify an existing user profile in the user profile list;
- 0 Delete a user profile from the user profile list; and
- 20 0 Select a mailbox maintenance functions.

Messages Sent from a Mailbox

- 25 This is a mailbox maintenance function that lists the messages sent by each user in the VAE system. Using the messages sent list, the system administrator can search for a message in the list.

Messages Received in a Mailbox

- 30 This is a mailbox maintenance function that lists the current messages received by a user in a given mailbox. Using this function, the system administrator can:
- 0 Search for a message in the list;
 - 0 Delete selected messages from the mailbox; and
 - 0 Clear all messages from the mailbox.

35 VOICE APPLICATION GENERATOR

The Voice Application Generator is the tool used for the Voice Application Enabler (VAE) application generation. It is a graphics workstation typically used by the application developer to create, modify, and customize voice applications. The VAG main functions are:

- 40 0 create and modify voice segments;
- 0 create prerecorded system messages known as prompts;
- 0 create state tables that promote the application flow and define edge and default conditions;
- 0 maintain and test voice applications;
- 0 provide links between the 3270 Server and other Host-based applications; and
- 45 0 build and implement the custom server.

Through the VAG facility, the application developer creates the original default scripts and prompts when the VAE system is first implemented. This function enables the creation of applications that allow the VAE to function, among its many other application capabilities, as an answering machine, voice messaging system, and voice response unit.

- 50 The VAG incorporates a SEARCH function and an on-line HELP system. The SEARCH function provides a list of search actions for all appropriate VAG routines. The HELP system provides:
- 0 menu-driven, context-sensitive help panel for each menu; and
 - 0 An item-level help screen for selected items on each menu.
- A display technique that emphasizes selected words in the help text is also provided. The displayed help text is language dependent. The VAG is multi-lingual, menu-driven, and consists of the:
- 55 0 Application Manager;
 - 0 Voice Generator;
 - 0 Prompt Generator; and

O State Generator.

Application Manager

5 The Application Manager maintains the applications that are available in the VAE system. It also manages the state tables, prompt directories, and voice segments that define an application. The Application Manager allows the user to perform the following functions:

- O Select an application from the application list;
- O Delete an existing application and all its component parts;
- 10 O Delete a state table;
- O Delete a prompt directory;
- O Debug a state table;
- O Provide a custom server interface;
- O Save a complete application to archive media; and
- 15 O Restore a complete application from archive media.

STATE TABLE DEBUGGER:

20 The state table debugger provides the functions that allow the system administrator to verify the functionality of a given state table, and to determine if and where problems exist within a given state table.

CUSTOM SERVER USER INTERFACE:

25 The Custom Server User Interface (CSUI) is the tool used for creating, building and maintaining custom servers. It is a graphics workstation typically used by the application developer to develop the Application Specification File. This Application Specification File is the module GPS uses to build the custom server. From the custom server application main screen, the application programmer can elect to browse or modify an existing custom server, or create a new server.

30 Voice Generator

35 The Voice Generator is used to create/modify/delete the basic unit of voice. This basic unit of voice is a word, phrase, sentence, or set of sentences and is called a voice segment. There is both a textual representation and an audible representation of the voice segment. The voice segment identifier is the link between the text and the audible voice segment.

The Voice Generator is an interactive program that allows the application developer to:

- O Create, modify, delete, and display the textual representation of the words, phrases and sentences recorded as voice segments; and
- O Create, modify, delete, and display the digitized voice segments.

40 The Voice Generator program is divided into two main parts: text editing and voice editing. Text editing consists of editing voice segments in the form of text, while voice editing consists of editing the audible voice segments in the form of digitized voice signals. The Voice Generator user interface consists of two work panels where the text and the digitized voice data are entered.

45 The first panel, illustrated in Figure 7, is the VAG Prompt Segment Editor. This display provides the user with a highly visual, user-friendly method of listing and maintaining the textual representation of the voice segments. Using a mouse and keyboard, the user can enter and modify textual representations of voice segments in a simple and straightforward manner. Then, by simply clicking a mouse button when the cursor is positioned on MOD VOICE, the user can change the panel from the textual representation of the voice segment to the digitized voice signal panel. In fact, a major feature of the Voice Generator is the ease in which the user can go from the text to the digitized version of the voice segment, thus working on both versions at the same time.

50 The second panel is a VAG Digitized Voice Editor panel similar to the panel illustrated in Figure 7. It allows the user to:

- O Select and display one or two voice segments;
- O Display all or selected portions of a voice segment using a zoom action, magnify the digitized voice signal for a closeup view;
- O Create new voice segments from existing voice segments;
- O Delete selected portions of a voice segment;
- O Copy selected portions of a voice segment to different places in the voice segment;

- ☐ Copy selected portions of one voice segment into another voice segment;
- ☐ Switch back and forth between two windows to edit two voice segments simultaneously; and
- ☐ Playback and record a voice segment.

5 Prompt Generator

The Prompt Generator is used to create/modify/delete prompts. These prompts are the recorded sentences or set of sentences that are presented to the subscriber when she communicates with the telephony system. The Prompt Generator is an interactive program that allows the application developer to:

- 10 ☐ Create, modify, delete, and display prompt directories; and
- ☐ Create, modify, delete, and display individual prompts.

The Prompt Generator program is divided into two main processes: defining prompt directories and defining the prompts that are listed in the directories. The Prompt Generator user interface consists of three primary work panels where the information necessary to define the prompts and prompt directories are entered.

- 15 The first and second panels are the VAG Prompt Directory Editor and the VAG Prompt List Editor panels. These user interface panels provide the user with a highly visual, user-friendly method of listing and maintaining both the prompt directory and the prompts themselves. Using a mouse and keyboard, the user can enter and modify the prompts and prompt directories in a simple and straightforward manner.

- 20 The third panel is the Prompt panel, illustrated in Figure 8, which is essentially the work area where the prompts are created. It provides the application developer with a list of voice segments 510 in which to build a prompt and the tools to create conditional tests within a prompt. It also provides a list of variables 500 when variables must be played in a prompt. These tools are presented as dialog windows that display the required information. All the user has to do is select the necessary information from the dialog windows. This user interface design provides a highly convenient and efficient way to build prompts from voice segments, variables, and conditional tests.

State Generator

- 30 The State Generator is a tool used to create/modify/delete state tables and states. A state is one stage or step in a logical sequence of actions that comprises a telephony application. A state table is a table comprised of these states. A state table provides the VAE with the basic rules to run the application through states, actions, parameters, and edge values.

- The State Generator is an interactive program that creates and updates state tables and states. It consists of the VAG State Table Editor and VAG State Editor panels. The VAG State Editor panel is illustrated in Figure 9. These panels allow the user to:

- 35 ☐ View the complete state table including the state number and purpose;
- ☐ Create, add, modify, or delete a state table;
- ☐ Select a state by number from the list displayed and view the details on the second panel;
- ☐ Set default options such as preassigned edge values;
- 40 ☐ Display a complete list of actions and their rules, such as inputs needed that include parameters and required edges
- ☐ Modify an existing state;
- ☐ Delete an existing state; and
- ☐ Create a new state.

- 45 When modifying or adding a state, the user can select an action from the list of displayed actions or by entering the action in the field provided on the panel. The parameters, if required, are also selected from a list displayed on the panel 600. Then, the State Generator prompts for the edges 610 that are required by the selected action.

- When the application is complete, a function called check consistency is enabled. Check consistency monitors:

- 50 ☐ The validity of the edges that reference an existing state;
- ☐ The validity and existence of the parameter(s), if needed. For each prompt, a test is performed in all defined languages to verify the existence of this prompt; and
- ☐ The minimum of logic required for a state, such as the existence of a CloseSession action.

55

VAE APPLICATION ACTIONS

The following is a collection of the actions that are used when defining a state table.

O Idle (Internal Action)
 O AnswerCall (Internal Action)
 O EndCall (Internal Action)
 O PlayPrompt
 5 O GetKey
 O GetData
 O GetText
 O GetFindName
 O GetFindPassword
 10 O EvaluateData
 O AssignData
 O PlayVoice
 O RecordVoice
 O SaveVoice
 15 O DeleteVoice
 O CheckStorage
 O CheckMailbox
 O UpdateMailbox
 O UpdateUserProfile
 20 O SendData
 O ReceiveData
 O GetFindData
 O CallStateTable
 O ExitStateTable
 25 O Disconnect
 O CloseSession
 Each of these actions are discussed below in sections bearing the name of the action.

Idle (Internal Action)

30 The Idle action is reserved for the internal state table only. This action is run by a Channel Process when it has nothing to do. It causes the process to sleep while waiting on a semaphore to be notified by the Node Manager. When notified, Idle determines whether the Node Manager is requesting the process to answer a call or place a call, and then returns the appropriate edge to the state machine.

35 PARAMETERS NONE.

EDGES

0 = AnswerCall;
 1 = PlaceCall; and
 HUP = EndCall.

40

AnswerCall (Internal Action)

The AnswerCall action is reserved for the internal state table only. This action initializes the Channel Process to process an incoming call. It retrieves the user profile for the called telephone number, the state table defined in the user profile, and the prompt directory defined in the state table. Then it answers the phone and evokes the State Machine with the state table and the starting edge that is defined in the user profile.

45 PARAMETERS NONE.

EDGES EndCall.

50 EndCall (Internal Action)

The EndCall action is reserved for the internal state table only. This action cleans up when a session is complete. It ensures that all lines used by this session have been reset (placed on-hook) and reassigns them in the line table back to the Node Manager.

55 PARAMETERS NONE.

EDGES Idle.

PlayPrompt

This action builds and plays the voice segments that are defined in the prompt directory. Prompts are played in the language specified in the application profile. Prompts consist of:

- 5 O The prompt id number. This number identifies the prompt and is used as the PlayPrompt parameter.
- O The force play option. This number indicates whether or not a prompt is force played.
- O The time out option. This number specifies the number of seconds the user has to respond at the next GetKey or GetData action.
- O The repeat option. This specifies the number of times a prompt is repeated before a T2 time out. A T2 time out initiates a Disconnect action.
- 10 O A list of segment ids, system variables, and conditional tests. The conditional test controls what segments and variables are to be played for a particular prompt based on conditions at run-time.

For example, PlayPrompt is used any time the system interacts with the user to give information and directives, or to answer questions. For instance, in a voice messaging system, if the user selects the LISTEN option, the prompt might be: "You have no new messages. You have four old messages. Your oldest message is two weeks old."

PARAMETERS

PROMPT NUMBER. There is no default.

FORCE PLAY. To force play means to play a prompt completely even when interrupted by a keyed input. Force play is used if there is an important message, such as voice storage is full. In this case, the message is not interrupted. The force play flag exists in the prompt directory. In this case, the force play parameter overrides the flag set in the prompt directory. The default is do not force if a key on the phone pad is pressed. This means that the user can interrupt the prompt by his keyed input if he does not want to listen to the prompt in its entirety. The system then automatically goes to the next step of the application.

EDGES

0 = The prompt has played completely except in the case where a prompt was interrupted by a keystroke. In this case, the prompt does not play completely.

1 = There is a voice channel problem.

30 **HUP** = The caller has hung up.

GetKey

Getkey is used to receive a keyed input from the caller when a choice of options was given in the previous prompt. The previous prompt also provides GetKey with the number of seconds to wait before time out occurs and the maximum number of times to repeat this prompt before time out. The GetKey action recognizes a single keyed input only.

For example, if the prompt is: "To record, press 1; to listen, press 3; to change personal options, press 8; to transfer out of the system, press 7, then press #", the logical state processed after the above PlayPrompt action is the GetKey action. In this example, key 1 activates a record session, key 3 activates a listen session, key 8 activates the personal options session, key 7 transfers out of the system, while all other keys execute a PlayPrompt stating, "I do not understand this command. Please try again."

PARAMETERS **BUFFER NAME.** The keyed input is stored in this buffer for future use. The default is use the key for the edge value only.

EDGES

0 = User pressed key 0

1 = User pressed key 1

2 = User pressed key 2

3 = User pressed key 3

5 = User pressed key 5

7 = User pressed key 7

9 = User pressed key 9

4 = User pressed key 4

6 = User pressed key 6

8 = User pressed key 8

* = User pressed key *

= User pressed key

T1= Time out

T2= Last repeat time

HUP= The caller has

hung up.

5

GetData

Where the GetKey action is used for a single key input, the GetData action enables the application to receive several keys in a single state step. When completing the input, the last key pressed must be the # key. This action accepts the keyed inputs and stores them in a variable. An edge is returned to reflect the status of the input. The previous prompt also provides GetData with the number of seconds to wait before time out occurs and the maximum number of times to repeat this prompt before time out.

For example if the prompt is: "Please enter your new password. It must be from five to eight characters long", the GetData action requires a keyed input that is a minimum length of five characters and a maximum length of eight characters. The caller must enter a password of from five to eight characters long, followed by pressing the # key.

PARAMETERS**PARM 1:** Buffer name in which the input is stored

PARM 2: If the buffer is a character string, this is the minimum length of the input. If the buffer is numeric, this is the minimum value of the input

PARM 3: If the buffer is a character string, this is the maximum length of the input. If the buffer is numeric, this is the maximum value of the input

EDGES

0 = Input length or value is correct

1 = Input length too short or value too small

2 = Input length too long or value too large

3 = Input incomplete. The # key was not pressed.

T1 = Time out. Nothing was entered.

T2 = Last repeat time

HUP = The caller has hung up.

GetText

The GetText action works much like the GetData action. This action enables the application to receive ASCII text data as input from the DTMF keypad. Two DTMF keys are pressed by the caller to designate a single ASCII character. When entry is completed, the caller must press the # key.

The ASCII text entered during this action is stored in a character buffer. An edge is returned to reflect the status of the entry: too short, too long, or time out occurred while waiting for the input.

PARAMETERS**PARM 1:** Buffer name where the input is to be stored;**PARM 2:** Minimum length of the input; and**PARM 3:** Maximum length of the input.**EDGES**

0 = Input successful

1 = Input too short

2 = Input too long

3 = Input incomplete, time out occurred

T1 = Timeout, no input yet

T2 = Last repeat time

HUP = The caller has hung up.

GetFindName

This action is used any time a caller or a message receiver must be identified by digit name or extension number. A digit name is a representation of the user's last name when it is spelled using the alphanumeric key pad on the telephone. GetFindName is used when the caller logs on to system services or when the caller sends messages through system services.

An illustrative embodiment of the present invention concerns a connected-word and -digit (hereinafter "connected-word") speech recognizer for information services. The embodiment exploits the idea that user speech in the information service context is often predictable, for example, from past speech of the user, or from constraints on or the nature of a request for information. Via a training or initialization procedure, one or more lists (*i.e.*, databases) of connected-word speech are built and maintained. A list comprises the likely spoken responses to a *given* request for information by the information service. For each connected-word speech recognition task, recognition is performed in the first instance by reference to the list or set of likely responses to that request. The unknown connected-word speech is compared to the entries in the list by assembling for each list entry appropriate reference patterns (as specified by each list entry) and by using a time alignment procedure such as Dynamic Time Warping. Each comparison to a list entry yields a comparison score. The unknown speech is recognized as the list entry with the best score below a user specified or machine determined threshold. For those occasions when no comparison score is below the threshold (or when two or more scores are below the threshold), one or more back-up procedures are provided.

Brief Description of the Drawings

Figure 1 presents an illustrative tree structure of a user interface for an information service.

Figure 2 presents an illustrative embodiment of the present invention.

Figure 3 presents a speech recognizer as an illustrative embodiment of the present invention.

Figure 4 presents an illustrative data structure for a list stored in the memory of the recognizer presented in Figure 3.

Figure 5 presents an illustrative data structure for word patterns stored in the memory of the recognizer presented in Figure 3.

Figure 6 presents an exemplary sequence of feature vectors as specified by an exemplary list response and associated word patterns presented in Figures 4 and 5, respectively.

Figures 7 and 8 present a flow chart of an illustrative process executed by a processor of the recognizer presented in Figure 3.

Figure 9 presents an illustrative graph of a Dynamic Time Warping alignment path, $w(n)$.

Figure 10 presents an illustrative embodiment of a connected-digit speech recognizer for a telephone repertory dialer.

Figures 11 and 12 present a flow-chart of the operation of the processor of the illustrative embodiment presented in Figure 7.

Detailed Description

Generally, user interfaces for information services operate according to a logical tree structure. Figure 1 presents a diagram of such a tree 10. The tree 10 includes nodes 15, branches 20, and tasks 25. Each node 15 represents an explicit or implicit request for information put to the user by the information service. Each node 15 is related to other nodes 15 by one or more branches 20. Each task 25 represents a function performed by the service for the user. As such, a series of requests made and responses given defines a logical path through nodes 15 and branches 20 of the tree 10 specifying a task 25 to be performed. Since each node 15 represents a request for information, each node 15 may also represent a task of resolving uncertainty in a response.

Figure 2 presents an illustrative embodiment 50 of the present invention. The embodiment 50 provides a comparator 51 and database 52. Database 52 comprises one or more likely responses to one or more requests for information (represented by nodes 15) put to an information service user. Information 53 is received from a service user via an input device in response to a service request and is provided to the comparator 51. To resolve uncertainty in received information 53, the comparator 51 provides control/data signals 55 to scan the database 52 for likely responses 56 associated with the request 54 which provoked user response information 53. The comparator 51 compares each likely response 56 from database 52 with the received information 53 to determine which likely response 56 most closely corresponds to the received response 53. (Alternatively, the comparator 51 may tentatively identify the received response 53 as the closest likely response 56 and wait for some user interaction concerning a right of refusal; or, the comparator 51 may identify the received response 53, tentatively or otherwise, as the first likely response 56 associated with the request encountered in the database 52 with a measure of similarity within a range of acceptable similarity scores.)

The comparator 51 outputs the determined likely response as the identified response 57.

A Speech Recognizer

Figure 3 presents a connected-word speech recognizer as a further illustrative embodiment of the present invention. The recognizer 100 comprises input device 101 (e.g., a microphone of an I/O device), an analog-to-digital (A/D) converter 102, processor 103, and memory 104. Memory 104 stores, among other things, one or more lists of likely responses to a request for information associated with a given node 15. Also shown in Figure 3 is a utilization device 105 to receive the response corresponding to the recognized speech. This utilization device 105 represents an information service. A bus 106 interconnects the A/D converter 102, the processor 103, the memory 104, and the utilization device 105. The A/D converter 102, processor 103, memory 104, and utilization device 105 may be located locally to the input device 101. Alternatively, one or more of these may be located at some distance and coupled to the local devices by a network.

Prior to considering the operation of the illustrative embodiment of Figure 3, it will be instructive to consider the contents of memory 104 as they concern a list and associated word patterns for recognizing speech.

The illustrative speech recognizer presented in Figure 3 exploits the idea that a request for information by an information service often provokes a spoken response which is predictable, for example, from past recognized (or "decoded") speech of the user or from constraints on or the nature of the request for information. Via one or more techniques discussed below, a list of likely responses to a given request for information is determined and stored in memory 104. Each likely response in the list comprises a series of one or more *references* to word patterns (e.g., word templates or statistical models) stored separately in memory 104. Each word pattern represents a word used in a likely response. A multiple-word likely response therefore comprises references to multiple word patterns.

Each word pattern stored in memory 104 comprises or is based on one or more speaker-independent or -dependent feature vectors. The feature vectors of a word pattern represent the salient spectral properties of the word in question. One type of feature vector comprises a mean of one or more spectral vectors, each of which is derived from a time-aligned slice (or "frame") of a sample (or "token") of given speech. For example, each feature vector may represent a 45 msec. frame of speech (*i.e.*, a 45 msec. slice of a word), with adjacent frames separated by 15 msec. on center. Together, feature vectors for successive frames form a word pattern "template." Another type of feature vector includes a mean and covariance of a grouping of successive spectral vectors in a given token, determined over several tokens. Such means and covariances are used in statistical models of speech, such as the hidden Markov model (HMM) known in the art.

Feature vectors (for templates or statistical models) for a given word pattern may be obtained with any of several feature vector measurement techniques well known in the art, for example, Linear Predictive Coding. For a discussion of feature measurement techniques, see L.R. Rabiner and S.E. Levinson, *Isolated and Connected Word Recognition - Theory and Selected Applications*, Vol. Com-29, No. 5, I.E.E.E. Transactions On Communications, 621-59 (May 1981); *see also*, L.R. Rabiner and R.W. Schafer, *Digital Processing of Speech Signals*, 396-455 (1978).

Illustrative data structures concerning list and word pattern storage in memory 104 are presented in Figures 4a and 4b. As shown in Figure 4a, a list comprises V likely responses to a given request for information (such that the list is indexed by v , $1 \leq v \leq V$). Each likely response (or list entry), R_v , comprises a certain number, $L(v)$, of *references* to word patterns also stored in memory 104 (such that each likely response, R_v , is indexed by l , $1 \leq l \leq L(v)$, and each $R_v(l)$ references a particular word pattern in memory 104).

As shown in Figure 4b, word pattern storage comprises W word patterns (such that the storage is indexed by w , $1 \leq w \leq W$) which are used in forming the responses of an associated list. Each word pattern, P_w , comprises a certain number, $J(w)$, of feature vectors (such that each pattern, P_w , is indexed by j , $1 \leq j \leq J(w)$), and each $P_w(j)$ references a particular feature vector in a word pattern.

A given response or list entry, R_v , can therefore be represented as a sequence of feature vectors, $S_v(m)$, the sequence determined by the sequence of word patterns, P_w , specified by the response, R_v , and the sequence of feature vectors, $P_w(j)$, forming each word pattern. Thus, a given response or list entry comprises $M(v)$ feature vectors $S_v(m)$, $1 \leq m \leq M(v)$.

Figure 4c presents an exemplary sequence of feature vectors, S_v . The sequence presented, S_4 , is that specified by response or list entry R_4 , which references word patterns P_2 , P_5 , and P_4 , respectively, as shown in Figures 4a and 4c. Each of the referenced word patterns comprises feature vectors as specified in Figure 4b. Figure 4c shows a sequence of 12 feature vectors ($M(4) = 12$) which make up the string, S_4 .

The operation of the illustrative embodiment of Figure 3 may now be discussed with reference to Figure 5. Figure 5 presents a flow chart 200 showing an illustrative process executed by processor 103 of the recognizer 100. Responsive to receiving a START signal from the utilization device 105 over bus 106, processor 103 begins its process by checking for the receipt of a digital version of unknown speech to be recognized (see Fig. 5, 210). Unknown speech is received by input device 101 and provided to the A/D converter 102 as analog

signal input, $s(t)$. The A/D converter 102 provides a digital signal version of the unknown speech, $s(k)$.

Once $s(k)$ is available, the processor 103 performs spectral feature measurement processing on the digital signal, $s(k)$, to produce a series of feature vectors, $T(n)$, of received information. The received information feature vectors are referred to as the "test pattern," where n indexes the individual feature vectors of the pattern.

The feature vectors are obtained with the same technique as employed in generating the feature vectors of the word patterns stored in memory 104 (e.g., Linear Predictive Coding), and have the same frame duration and frame spacing. Feature vectors, $T(n)$, are representative of salient spectral properties of the unknown speech signal, $s(t)$. Thus, the test pattern may be categorized as received information. Test pattern feature vectors, $T(n)$, are stored in memory 104 (see Fig. 5, 220).

To recognize a test pattern of unknown speech, the processor 103 compares the test pattern to each of the V likely responses contained in the appropriate list for the request. Each comparison takes into account the similarity of the feature vectors of the test pattern, $T(n)$, and those feature vectors, $S_v(m)$, formed by a series of one or more word patterns specified by a likely response in the list. The comparison is made by a technique known in the art as dynamic time alignment.

Assuming the list contains one or more likely responses (see Fig. 5, 230), the processor 103 begins the time alignment process with the series of word patterns of the first likely response in the list, $R_1(l)$, for $l \leq 1 \leq L(1)$. (see Fig. 5, 235). Time alignment is performed between the test pattern feature vectors, $T(n)$, and a sequence of feature vectors, $S_1(m)$, formed by the series of word patterns specified by the first likely response, R_1 (see Fig. 5, 240; see also the *Dynamic Time Alignment* section below and Figure 6). A comparison score, D_1 , indicating the similarity or distance of the likely response with the test pattern is generated and saved (see Fig. 5, 245). The process is repeated for each of the likely responses in the list, R_v , $2 \leq v \leq V$. As a result, a set of comparison scores, D_v , $1 \leq v \leq V$ (see Fig. 5, 250) is determined. The list response which yields the best comparison score, D^* , below a threshold is deemed to be the recognized response, R^* (see Fig. 5, 255, 260).

The threshold value may be set arbitrarily or as part a training procedure for words in pattern storage. A typical value for the threshold corresponds to one standard deviation (1σ) of word pattern or "token" comparison scores above a mean comparison score determined during a training process for word patterns stored in memory 104 (see discussion of training in *List and Word Pattern Storage* section below).

If the comparison score, D^* , is below the threshold (meaning that a *good* recognized response has been found), the recognized response, R^* , is output to the utilization device (information service) 105. If desired, the comparison score, D^* , may be output as well (see Fig. 5, 260 and 280).

If the comparison score, D^* , is not below the threshold, or if the list does not contain any likely responses, one or more back-up procedures are used to recognize the speech. A response corresponding to recognized speech from a back-up procedure is then output to the utilization device (information service) 105 (see Fig. 5, 265, 270, 290). One back-up procedure which may be used comprises user manual entry of the information (see Fig. 5, 275). This may occur in response to a prompt of the user by the system via the I/O device. For a given embodiment, user manual entry may be the only back-up procedure needed.

Whether speech is recognized by the list or recognized or supplied by a back-up procedure, the list and pattern storage may be updated to incorporate statistics of response usage or to expand the list (in the case of back-up speech recognition) such that succeeding iterations of the speech may be recognized without resorting to a back-up scheme (see Fig. 5, 295). Thus, a "new" response may be added to the list as a set of references to stored word patterns, and test pattern information may be used to provide additional training for word patterns in pattern storage.

Also, as an option, an embodiment of the invention may provide the user with an opportunity to reject a recognized response. Under such circumstances, another automatic speech recognition process may be invoked or manual entry of the equivalent of spoken words can be performed.

Dynamic Time Alignment

The dynamic time alignment referenced above and in Figure 5, 240, can be accomplished by any of the techniques well-known in the art. An exemplary technique for performing one form of dynamic time alignment, namely Dynamic Time Warping (DTW) based on word templates, is discussed with reference to Figure 6 which presents a grid of points in a coordinate system. A sequence of feature vectors which make up the test pattern, $T(n)$, is mapped to the abscissa (the independent variable) (see, e.g., Figure 4c) and a sequence of feature vectors, $S_v(m)$, which make up a likely response, R_v , is mapped to the ordinate (the dependent variable). Each point in the grid represents the similarity or correspondence between the n^{th} feature vector $T(n)$ of the test pattern and the m^{th} feature vector $S_v(m)$ of the sequence of vectors of the likely response, R_v . A measure of similarity may be obtained according to the Itakura log likelihood ratio, as described in the article by F. Itakura

entitled, "Minimum Prediction Residual Principle Applied to Speech Recognition", *I.E.E.E. Transaction on Acoustics, Speech, and Signal Processing*, Vol. ASSP-23, No. 1, pages 67-72, February, 1975:

$$d(T(n), R_v(m)) = \log[T(n) \cdot S_v(m)] \quad (1)$$

i.e., a log of the dot product of the two vectors $T(n)$ and $S_v(m)$.

The quantity d is referred to as the "local distance" because the magnitude of d increases as the correspondence between $T(n)$ and $S_v(m)$ decreases (of course, other measures of similarity may be used, such as correlation coefficients which increase as the correspondence between $T(n)$ and $S_v(m)$ increases).

Since test pattern feature vector index n is defined to be the independent variable, the likely response feature vector index m may be written equivalently as a function of n , that is,

$$m = w(n), \quad (2)$$

where $w(n)$ represents a path through the grid as shown in Figure 6. The local distance, d , of equation (1) may therefore be written as $d(T(n), S_v(w(n)))$.

In order to optimally align the test pattern feature vectors with the sequence of feature vectors of a likely response, the sum of the local distance signals $d(T(n), S_v(w(n)))$ between the feature vectors of test pattern, $T(n)$ and the likely response, $S_v(w(n))$, is minimized:

$$D_v = \min_{w(n)} \frac{1}{N} \sum_{n=1}^N d(T(n), S_v(w(n))). \quad (3)$$

The quantity D_v is the comparison score (or global average distance) for a likely response, R_v . The likely response, R_v , $1 \leq v \leq V$, which yields the minimum comparison score, D^* , is the best candidate for identifying the input test pattern, $T(n)$.

In order to obtain a given comparison score, D_v , certain assumptions are made. First, it is assumed that the beginning and ending frames of both the input and reference words have been accurately determined. The first input frame $n=1$ is thus paired with the first reference frame $m=1$, or:

$$w(1) = 1. \quad (4)$$

Similarly, the last input frame $n=N$ is paired with the last reference frame $m=M$:

$$w(N) = M. \quad (5)$$

It is also assumed that the Itakura path constraints are obeyed:

$$0 \leq w(n) - w(n-1) \leq 2, \quad (6)$$

and

$$w(n) = w(n-1) \neq 0 \text{ if } w(n-1) - w(n-2) = 0. \quad (7)$$

These local path constraints guarantee that the average slope of the warping function $w(n)$ lies between $1/2$ and 2 , and that the path is monotonic non-decreasing. In other words, the local path constraints define acoustically reasonable and allowable paths.

The preceding endpoint and local path constraints may be summarized by a set of global path constraints:

$$m_L(n) \leq m \leq m_H(n) \quad (8)$$

where

$$m_L(n) = \min[2(n-1) + 1, M - \frac{1}{2}(N-n), M] \quad (9)$$

and

$$m_H(n) = \max[\frac{1}{2}(n-1) + 1, M - 2(N-n), 1] \quad (10)$$

The global path constraints define the parallelogram (or window) shown in Figure 6. Allowable paths include only points within the parallelogram.

The path $w(n)$ yielding a minimum distance or comparison score, D_v , can be found by a dynamic programming process. An accumulated distance, D_A , at any given pair of frames n and m is defined to be the sum of the local distances d from point $(1,1)$ to and including the present point (n,m) , along the minimum distance or "best" path between points $(1,1)$ and (n,m) . Accumulated distance D_A may be generated recursively from point $(1,1)$ to point (N,M) according to the following equation:

$$D_A(n,m) = d(T(n), S(m)) + \min[D_A(n-1, m)g(n-1, m) - D_A(n-1, m-1), D_A(n-1, m-2)], \quad (11)$$

where the constraints are

$$1 \leq n \leq N, m_L(n) \leq m \leq m_H(n) \quad (12)$$

and where $g(n,m)$ is a nonlinear weighting

$$\begin{aligned}
 g(n,m) &= 1, \text{ if } w(n) \neq w(n-1) \\
 &= \infty, \text{ if } w(n) = w(n-1),
 \end{aligned}
 \tag{13}$$

to guarantee that the optimum path to (n,m) does not stay flat for two consecutive frames. The desired comparison score, D_v , for a likely response, R_v , is thereby equal to the accumulated distance $D_A(N, M)$.

This procedure may be performed for each likely response, R_v , providing values for D_v , $1 \leq v \leq V$. The test pattern, $T(n)$, can be recognized as the likely response, R_v , with the minimum comparison score, D^* , smaller than a threshold for "good" scores.

List and Word Pattern Storage

As discussed above, the illustrative speech recognizer embodiment employs list and pattern storage in recognizing *likely* spoken responses to a request for information. The list comprises one or more likely responses, each comprising a string of one or more references to stored word patterns. Each word pattern referenced by a likely response comprises or is based upon one or more feature vectors derived from either speaker-independent or -dependent data (that is, based on speech tokens from multiple people or a single person, respectively). The contents of list and pattern storage may be determined from knowledge of likely user responses, from experience (*i.e.*, training) with a user, or both.

Knowledge of likely user responses is often derived from the associated request for information. Thus, list responses and word patterns may be determined based upon the nature of the request (*e.g.*, determined based upon the type of information sought) or the constraints placed on a response by the terms of the request (*e.g.*, by choices given to a service user from which to select as a response). For example, if a request were to ask a user to specify a color, the nature of the request would suggest a list which included the responses "red," "blue," "orange," etc., and supporting patterns. On the other hand, if a request to specify a color included a menu of alternatives - "red," "green," or "yellow" - then these choices should be in the list as likely responses with supporting patterns provided.

Knowledge of likely responses and associated patterns may also be obtained from the nature of the information service itself. For example, if an information service is concerned with taking orders for automobile parts, such words as "sparkplug," "muffler," "headlight," and "filter," among others, might be provided by a list and pattern storage.

A list of likely responses and supporting patterns may be provided through experience or training ("training") with a user. Such training generally requires either manual user action, or the use of other speech recognition techniques well-known in the art, such as a Vector Quantization Codebook scheme (see Linde, Buzo, and Gray, *An Algorithm for Vector Quantization Design*, Vol. Com-28, No. 1, I.E.E.E. Transactions on Communications, 84-95 (Jan. 1980)) or the "level building" technique of Myers and Rabiner (see Myers and Rabiner, *A Dynamic Time Warping Algorithm for Connected Word Recognition*, Vol. ASSP-29, I.E.E.E. Trans. Acoust., Speech, Signal Processing, 284-97 (Apr. 1981)). Training may be performed prior to recognizer use as part of a training mode, or during use in the form of one or more back-up procedures. Moreover, training provided by speech recognition techniques may be performed locally or off-line and provided to the system via, *e.g.*, a read-only memory.

Manually provided training may require a user to provide data equivalent to a spoken response through the use of an I/O device, such as a keyboard. This data is used to update the stored list. Manual training may also involve creating or updating patterns in word pattern storage by requiring a user(s) to speak samples (or tokens) of words one or more times. These samples, once processed by a feature measurement technique, are used, *e.g.*, to form one or more mean spectral vectors (*i.e.*, one or more feature vectors) for a word pattern. Each word pattern, P_w , is stored in pattern storage as a word to be referenced by list responses, R_v .

If a speech recognition scheme is used to provide training, the output of such a scheme may serve to augment the list and update word pattern storage. The list may be updated by including a newly recognized response as a likely response, R_v . Pattern storage may be updated by including recognized test pattern information in the computation of, *e.g.*, mean spectral vectors for word patterns.

Whether through knowledge or training, the determination of one or more likely responses reflects *a priori* probabilities that a given request will provoke such responses. If probable responses are known prior to information service use, then these probable responses can be provided to a list with supporting pattern storage. Regardless of whether any such responses are known prior to use, those likely responses determined through training (either during a training mode or with use) may augment the list and update pattern storage.

Referring to Figure 5, the selection of patterns of a response (see 235) is directed, at least initially, to those responses considered likely prior to training. However, if no responses are considered likely prior to training (see 230), or if the list of likely responses fails to produce a recognized response with a comparison score below the threshold for good recognized responses (see 260), one or more alternate procedures may be employed
 5 to perform speech recognition and provide the recognized speech to update the list and pattern storage (see 265, 270, 275, 295).

A Connected-Digit Repertory Dialer

10 A further illustrative embodiment of the present invention concerns a connected-digit speech recognizer for a telephone repertory dialer. With this embodiment, a user speaks a telephone number in a connected-digit fashion (i.e., fluently) in response to an explicit or implicit request and the speech is recognized and provided to an automatic dialer.

In this embodiment, a list is stored which comprises telephone numbers which are likely to be dialed by a user. Likely numbers comprise numbers which either are or will be frequently dialed. Each digit or group of
 15 digits of a likely number references a sequence of feature vectors in pattern storage.

A list of likely numbers may be built in any number of ways. For example, the list may be built manually through user entry of the likely numbers directly from a telephone keypad, either as part of a special mode providing for such entry (under the control of the processor), or as part of a back-up procedure when no list
 20 entry for the number exists. Also, the list may be built automatically by observation of normal telephone usage, either locally (i.e., at the telephone itself) or by a node(s) in a network to which the telephone is connected. Whether built manually or automatically, locally or by a network, the list containing likely telephone numbers may be stored locally or at an external network location.

The pattern storage comprises speaker-independent feature vectors for the words corresponding to the ten digits, zero through nine, and the usual associated words, such as "oh," "hundred," and "thousand." In
 25 addition, the pattern storage may include patterns for one or more user command words, such as "off-hook," "dial," "hang-up," "yes," "no," etc.

Pattern storage may also include patterns for one or more names of people, businesses or services likely to be called; that is, the names associated with likely numbers in the list. In this way, a number may be dialed
 30 by the illustrative embodiment either as a result of a user speaking digits or by speaking the name of the person, business or service to be called. A representation of a telephone number in a list may therefore relate to the number itself, an associated name, or both (in which case an association in list memory between number and name representations would be established). Telephone number information received from a user to be recognized may comprise a number or an associated name.

35 The illustrative embodiment of a connected-digit speech recognizer 300 for a telephone repertory dialer is presented in Figure 7. Telephone 301 serves as an I/O device used for entry of speech to be recognized. The telephone 301 comprises an automatic dialer which requires input of a telephone number from the speech recognizer 300. Thus, in this embodiment, the telephone 301 serves as the utilization device referenced in Figure 3. The telephone 301 is coupled to an analog-to-digital (A/D) and digital-to-analog (D/A) converter 302.
 40 The telephone 301 is also coupled to a processor 303 and memory 304 by bus 305. The A/D and D/A converter 302 is also coupled to bus 305, and thereby coupled to the processor 303 and memory 304. Processor 303 comprises a feature measurement processor and a dynamic time alignment processor. For a given illustrative embodiment, processor 303 may further comprise a back-up speech recognition processor, such as a VQC recognition processor.

45 The operation of the illustrative embodiment of Figure 7 is presented in the flow-chart 400 of Figure 8. Upon receipt of a START command from telephone 301, the processor 303 waits to receive a digitized version of a spoken telephone number to be dialed (see Fig. 8, 410). Contemporaneously, a spoken telephone number is received by the telephone 301 and provided to the A/D converter 302 which, in turn, provides the digitized version of the spoken number, $s(k)$, to the processor 303. Responsive to receipt of $s(k)$, the processor 303
 50 performs feature measurement on $s(k)$ to produce a series of feature vectors, $T(n)$ (see Fig. 8, 420) for storage in memory 304. Assuming the list contains one or more likely telephone numbers, (see Fig. 8, 430), DTW of $T(n)$ is performed with each number, R_v , in the list and a comparison score, D_v , is kept for each DTW performed (see Fig. 8, 435, 440, 445, 450).

The best comparison score, D^* , from all comparison scores for the list is determined (see Fig. 8, 455) and, if it is below a threshold (see Fig. 8, 460), the list entry corresponding to the best score, R^* , is deemed to be
 55 the telephone number spoken. Therefore, the number, R^* , is provided to the telephone 301 via bus 305 for dialing.

If the best score, D^* , is not below the threshold, or if the list contained no entries of likely numbers to be

dialed, alternative or back-up techniques for speech recognition are performed. For purposes of this illustrative embodiment, a first technique comprises Vector Quantization Codebook (VQC) recognition (see Fig. 8, 465). VQC recognition techniques are well known in the art. See, Pan, Soong and Rabiner, *A Vector-Quantization-Based Preprocessor for Speaker-Independent Isolated Word Recognition*, Vol. ASSP-33, No. 3, I.E.E.E. Transactions on Acoust., Speech, and Signal Processing, 546-60 (June 1985); see also U.S. Patent No. 4,860,385, which is hereby incorporated by reference as if set forth fully herein; see also Shore and Burton, *Discrete Utterance Speech Recognition Without Time Alignment*, Vol. IT-29, No. 4, I.E.E.E. Transactions on Information Theory, 473-91 (July 1980).

If the VQC recognition is successful (see Fig. 8, 470), the recognized telephone number is provided to the telephone 301 for dialing (see Fig. 8, 490).

If the VQC recognizer fails to recognize the spoken number (see Fig. 8, 470), then the user is prompted by this embodiment to dial the number manually (see Fig. 8, 475) with telephone 301.

As it concerns any speech recognition task (i.e., telephone numbers or commands), this illustrative embodiment may also provide a user with an opportunity to reject recognized speech. Under such circumstances, another technique (e.g., a back-up technique) or manual entry may be employed.

Regardless of how the number is dialed, information concerning the dialed number is used to update the list (see Fig. 8, 500). The update to the list may involve storage of a telephone number not previously stored therein such that future attempts at dialing the number may be recognized without resorting to a back-up procedure. It may also involve using test pattern information to update the training of feature vectors for word patterns. It may further involve storing information concerning the usage of the telephone number by the user, such as the number of times the telephone number has been dialed or the date of last dialing. Such usage information may be employed in a likely response comparison scheme wherein likely responses are arranged in order of likelihood and a received response is identified, tentatively or otherwise, as the first encountered response which yields an acceptable comparison score. Such usage information may be also used as a basis for dropping or replacing a number previously stored in the list (e.g., if storage space is limited).

Just as telephone numbers to be dialed may be recognized through storage in the list, so may command words which control overall recognizer function. So, for example, speaker-independent vector patterns for words such as "off-hook," "dial," "hang-up," "yes," "no," etc., may be included in pattern storage and referenced in the list to provide hands-free operation of a telephone incorporating this embodiment of the present invention. In this embodiment, the voice command "dial" may be recognized and used to prompt the processing of a spoken telephone number through the issuance of a START command.

Claims

1. A method for resolving uncertainty in information provided to an information service, the method comprising the steps of:
 - storing in a database a list of one or more likely responses to a request for information;
 - receiving information from a user of the information service in response to the request for information; and
 - comparing received information with one or more of the likely responses in the list to identify such received information.
2. The method of claim 1 wherein the step of storing a list of one or more likely responses in a database comprises the step of determining a likely response based on an *a priori* probability that a response will be provoked by the request.
3. The method of claim 2 wherein the step of determining a likely response comprises the step of determining an *a priori* probability based on training with a user.
4. The method of claim 3 wherein training is provided by a back-up procedure for resolving uncertainty.
5. The method of claim 2 wherein the step of determining a likely response comprises the step of determining an *a priori* probability based on the nature of the information service.
6. The method of claim 2 wherein the step of determining a likely response comprises the step of determining an *a priori* probability based on the nature of the request for information.
7. The method of claim 2 wherein the step of determining a likely response comprises the step of determining

an *a priori* probability based on constraints placed on responses by the request for information.

8. The method of claim 1 wherein the step of comparing received information to one or more likely responses comprises the step of determining a comparison score for a likely response.
- 5 9. The method of claim 8 wherein the step of determining a comparison score comprises the step of determining whether a comparison score is within a range of acceptable comparison scores to identify received information.
- 10 10. The method of claim 9 further comprising the step of performing a back-up uncertainty resolution technique when no comparison score is within the range of acceptable comparison scores.
11. The method of claim 10 further comprising the step of updating the list of stored likely responses with results of the back-up uncertainty resolution technique.
- 15 12. The method of claim 1 further comprising performing a back- up uncertainty resolution technique when two or more comparison scores are within the range of acceptable comparison scores.
13. The method of claim 1 further comprising the step of maintaining likely response usage statistics based on identified received information.
- 20 14. The method of claim 1 further comprising the step of the user rejecting an identification of received information.
- 25 15. The method of claim 14 further comprising the step of performing a back-up uncertainty resolution technique.
16. A method for speech recognition comprising the steps of:
storing in a database a list of one or more representations of likely spoken responses to a request for information;
receiving speech information from a user in response to the request; and
30 comparing received speech information to one or more representations of likely responses in the list to recognize such received speech information.
17. The method of claim 16 wherein the likely spoken responses comprise telephone numbers.
- 35 18. The method of claim 17 further comprising the step of dialing the likely telephone number corresponding to recognized speech information.
19. The method of claim 16 wherein the step of storing a list of one or more representations of likely spoken responses comprises the steps of:
storing one or more word patterns comprising one or more feature vectors; and
40 storing one or more references to stored word patterns as a representation of a likely spoken response.
20. The method of claim 19 wherein the step of storing one or more word patterns comprises the step of determining such word patterns with a feature measurement technique.
- 45 21. The method of claim 20 wherein the feature measurement technique comprises linear predictive coding.
22. The method of claim 19 further comprising the step of updating a stored word pattern with recognized received speech information.
- 50 23. The method of claim 16 wherein the step of storing in a database a list of one or more representations of likely spoken responses comprises the step of determining a likely spoken response based on an *a priori* probability that a spoken response will be provoked by the request.
- 55 24. The method of claim 23 wherein the step of determining a likely spoken response comprises the step of determining an *a priori* probability based on training with a user.
25. The method of claim 24 wherein training is provided by a back-up procedure for recognizing speech.

26. The method of claim 25 wherein the back-up procedure comprises vector quantization codebook speech recognition on received speech information.
- 5 27. The method of claim 25 wherein the back-up procedure comprises the user supplying an equivalent to the received speech information with use of an input device.
28. The method of claim 16 wherein the step of receiving speech information comprises the step of producing a test pattern of received information by a feature measurement technique.
- 10 29. The method of claim 28 wherein the feature measurement technique comprises linear predictive coding.
30. The method of claim 16 wherein the step of comprising received speech information to one or more likely spoken responses comprises the step of determining a comparison score for a likely spoken response.
- 15 31. The method of claim 30 wherein the step of determining a comparison score for a likely response comprises the step of performing dynamic time alignment between received speech information and a likely spoken response.
32. The method of claim 31 wherein the step of performing dynamic time alignment between received speech information and a likely spoken response comprises the step of performing dynamic time warping.
- 20 33. The method of claim 30 wherein the step of determining a comparison score comprises the step of determining whether a comparison score is within a range of acceptable comparison scores to recognize received speech information.
- 25 34. The method of claim 33 further comprising the step of performing a back-up speech recognition technique when no comparison score is within the range of acceptable comparison scores.
- 35 35. The method of claim 34 further comprising the step of updating the list of stored representations of likely spoken responses with results of the back-up speech recognition technique.
- 30 36. The method of claim 16 further comprising the step of maintaining likely response usage statistics based on recognized received speech information.
37. The method of claim 16 further comprising the steps of:
the user rejecting a recognition of received speech information; and
performing a back-up speech recognition technique.
38. The method of claim 37 further comprising the step of updating the list of stored representations of likely spoken responses with results of the back-up speech recognition technique.
- 40 39. The method of claim 16 further comprising the step of updating the list of stored representations of likely spoken responses with recognized received speech information.
- 45 40. The method of claim 16 wherein the step of comparing comprises the steps of:
comparing received speech information to each stored representation of a likely spoken response;
and
determining the likely spoken response whose representation most closely compares to received speech information.
- 50 41. An apparatus for resolving uncertainty in information received from an input device to be provided to an information service, the information received in response to a request for information, the apparatus comprising:
a database storing one or more responses to the request for information based on *a priori* probabilities that such responses will be provoked by the request; and
a comparator, coupled to the database and the input device, for comparing received information with one or more responses in the list to identify such received information.
- 55 42. A system for recognizing spoken telephone number information, the telephone number information received from an input device, the system comprising:

a database for storing one or more representations of telephone numbers likely to be spoken; and
 a comparator, coupled to the database and the input device, for comparing spoken telephone number information with one or more representations of stored telephone numbers to recognize such information as a stored representation of a telephone number.

5

43. The system of claim 42 further comprising an automatic dialer, coupled to the comparator, for dialing the telephone number associated with the recognized information.

10

44. The system of claim 43 wherein the coupling between the automatic dialer and the comparator comprises a network.

15

45. The system of claim 42 wherein the comparator comprises a feature measurement processor, coupled to the input device, for performing feature measurements on the spoken telephone number information.

46. The system of claim 45 wherein the comparator further comprises a dynamic time alignment processor, coupled to the database and the feature measurement processor, for performing dynamic time alignment between feature measurements of the spoken telephone number information and one or more representations of stored telephone numbers.

20

47. The system of claim 42 wherein the database storing one or more representations of telephone numbers comprises:

one or more word patterns comprising one or more feature vectors; and

one or more references to stored word patterns as a representation of a likely spoken telephone number.

25

48. The system of claim 42 further comprising a back-up speech recognizer for recognizing spoken telephone number information when the comparator does not recognize such information.

49. The system of claim 48 wherein the back-up speech recognizer comprises a vector quantization codebook recognizer.

30

50. The system of claim 42 wherein the coupling between the database and the comparator comprises a network.

51. The system of claim 42 wherein the coupling between the input device and the comparator comprises a network.

35

52. A database for use with a speech recognition system coupled thereto, the database comprising one or more likely responses to a request for information, each such likely response having associated therewith an *a priori* probability that the response will be provoked by the request.

40

53. The database of claim 52 wherein the coupling of the database and the speech recognition system comprises a network.

45

50

55

FIG. 1

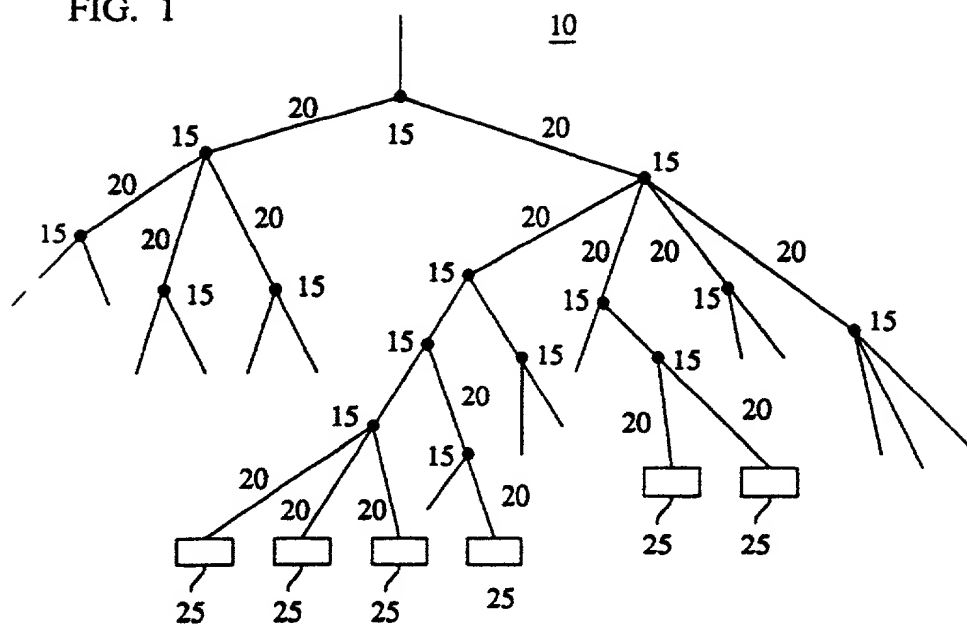


FIG. 2

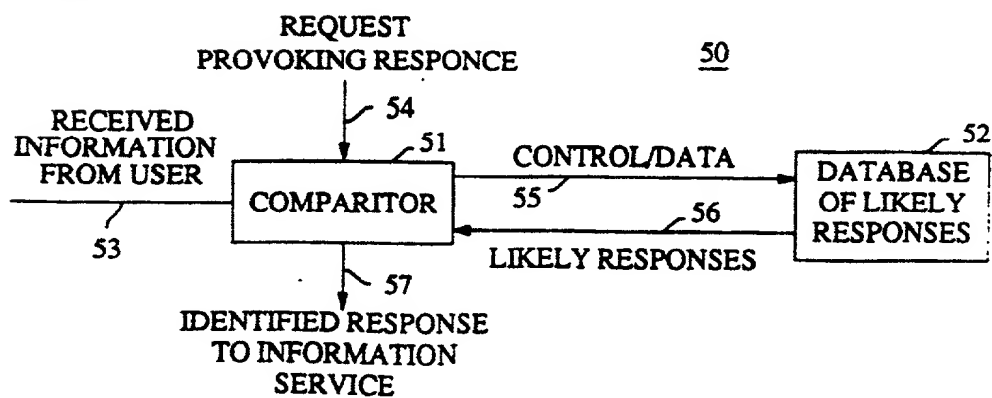


FIG. 3

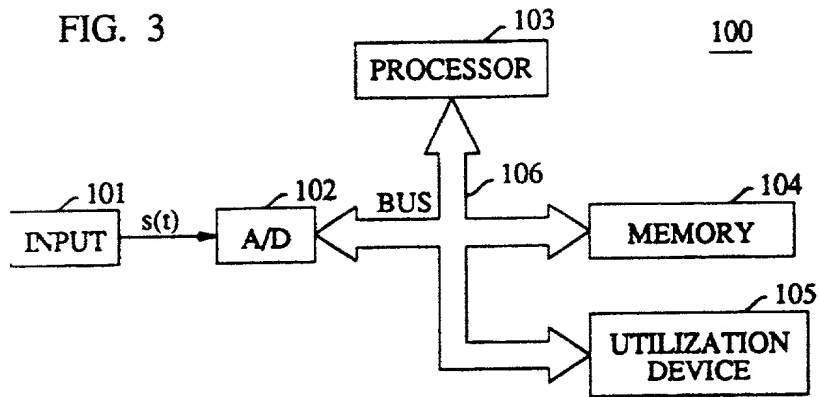


FIG. 4

LIST ENTRY

R_1	$R_1(1)$	$R_1(2)$	$\bullet \quad \bullet \quad \bullet$		$R_1(L(1))$
R_2	$R_2(1)$	$R_2(2)$	$\bullet \quad \bullet \quad \bullet$		$R_2(L(2))$
R_3	$R_3(1)$	$R_3(2)$	$\bullet \quad \bullet \quad \bullet$	$R_3(L(3))$	
R_4	$R_4(1)$ (= P2)	$R_4(2)$ (= P5)	$R_4(3)$ (= P4)		
R_5	$R_5(1)$	$R_5(2)$	$\bullet \quad \bullet \quad \bullet$		$R_5(L(5))$
\bullet	\bullet				
\bullet	\bullet				
\bullet	\bullet				
R_V	$R_V(1)$	$R_V(2)$	$\bullet \quad \bullet \quad \bullet$		$R_V(L(V))$

FIG. 5

PATTERN
ENTRY

P_1	$P_1(1)$	$P_1(2)$	• • •		$P_1(J(1))$ ($J(1)=5$)	
P_2	$P_2(1)$	$P_2(2)$	$P_2(3)$	$P_2(J(2))$ ($J(2)=4$)		
P_3	$P_3(1)$	$P_3(2)$	• • •			$P_3(J(3))$ ($J(3)=6$)
P_4	$P_4(1)$	$P_4(2)$	$P_4(3)$ ($J(4)=3$)			
P_5	$P_5(1)$	$P_5(2)$	• • •		$P_5(J(5))$ ($J(5)=5$)	
•			•			
•			•			
•			•			
P_W	$P_W(1)$	$P_W(2)$	• • •			$P_W(J(W))$ ($J(W)=6$)

FIG. 6

$$R_4 = R_4(1), R_4(2), R_4(3)$$

$$R_4(1) = P_2$$

$$R_4(2) = P_5$$

$$R_4(3) = P_4$$

	$R_4(1)=P_2$				$R_4(2)=P_5$					$R_4(3)=P_4$		
$S_v(m)$	$P_2(1)$	$P_2(2)$	$P_2(3)$	$P_2(4)$	$P_5(1)$	$P_5(2)$	$P_5(3)$	$P_5(4)$	$P_5(5)$	$P_4(1)$	$P_4(2)$	$P_4(3)$
$m=$	1	2	3	4	5	6	7	8	9	10	11	12

$$v=4$$

$$M(v)=12$$

FIG. 7

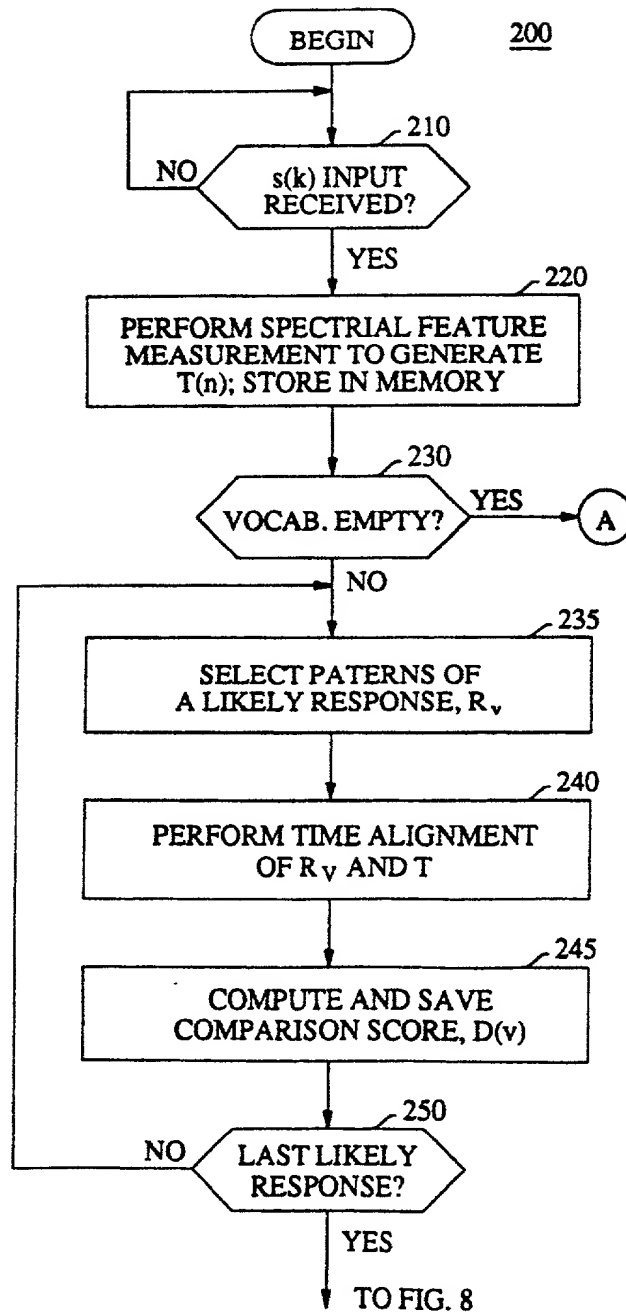


FIG. 8

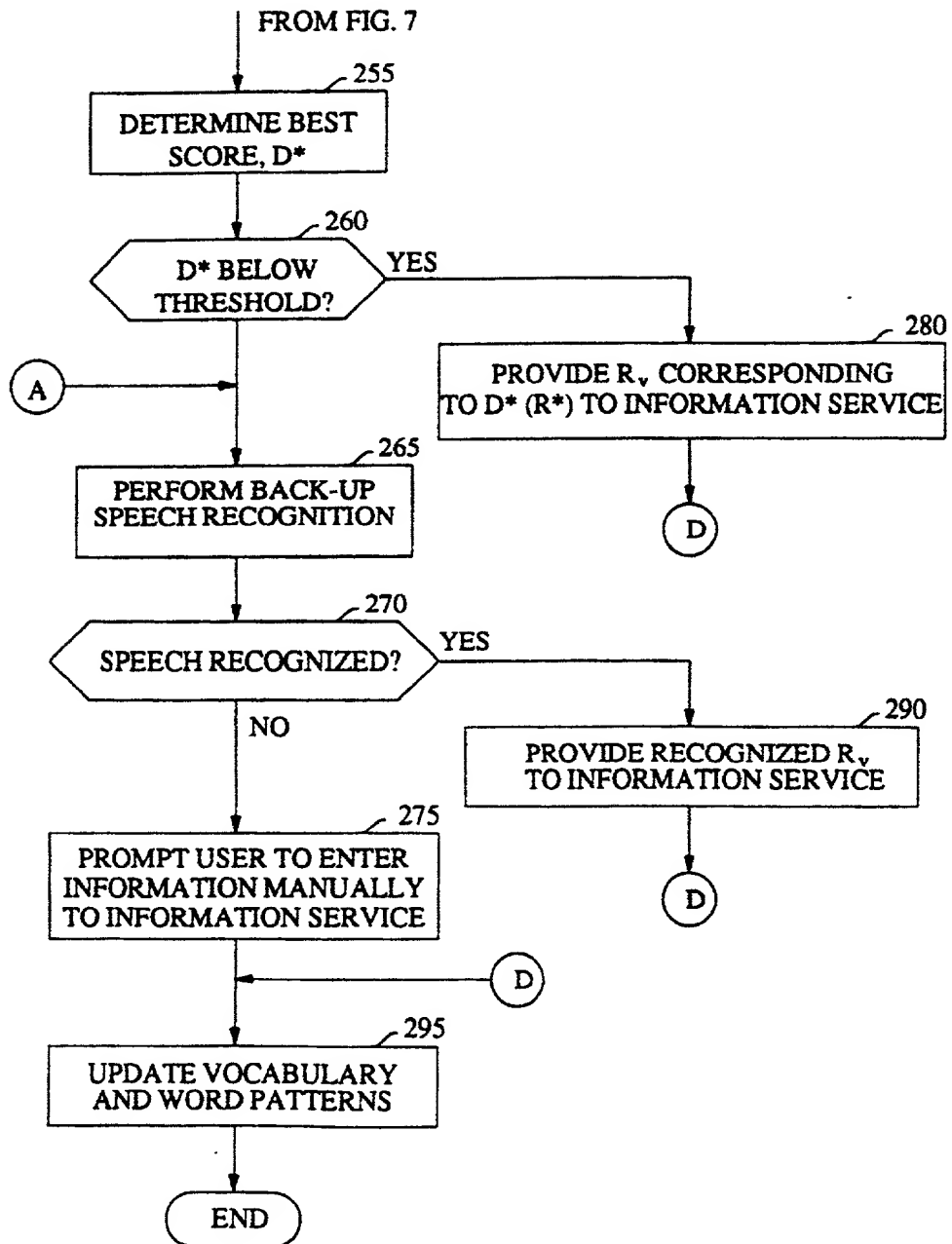


FIG. 9

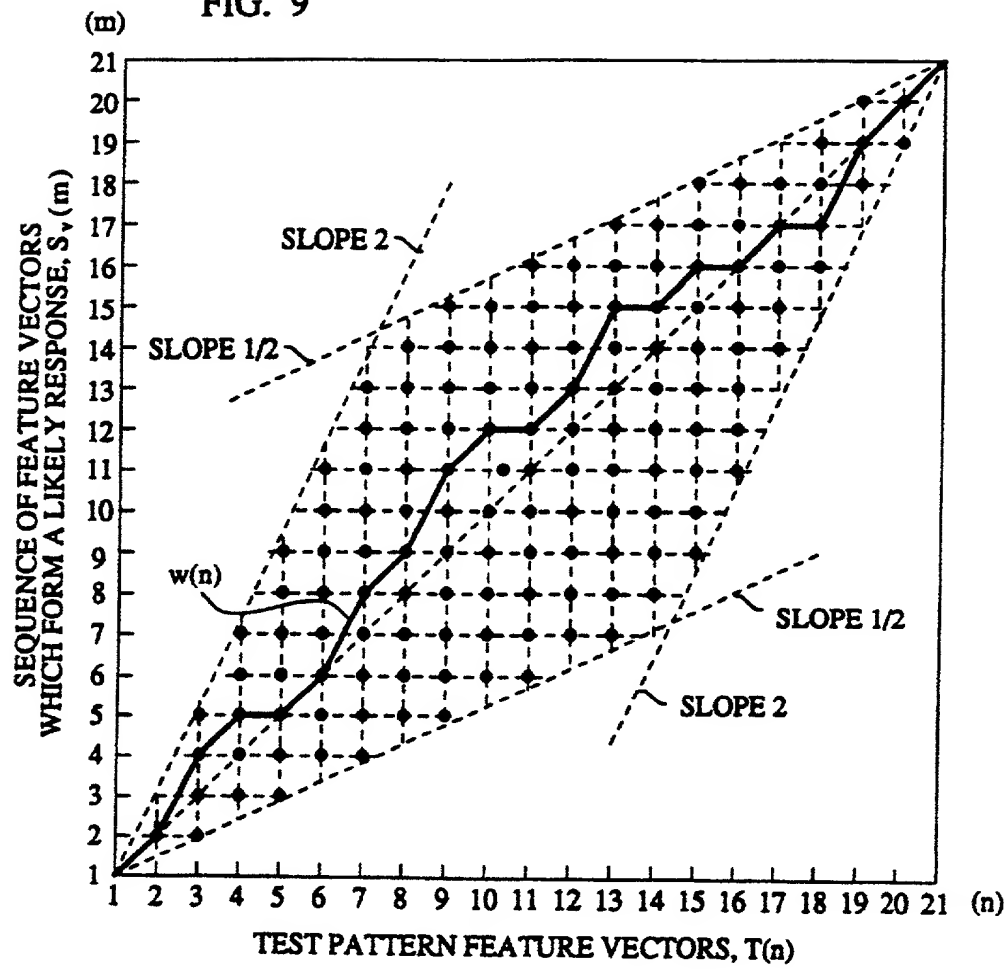


FIG. 10

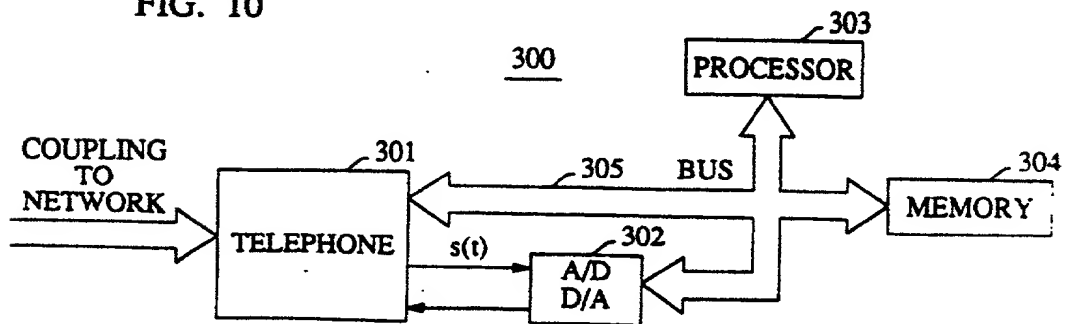


FIG. 11

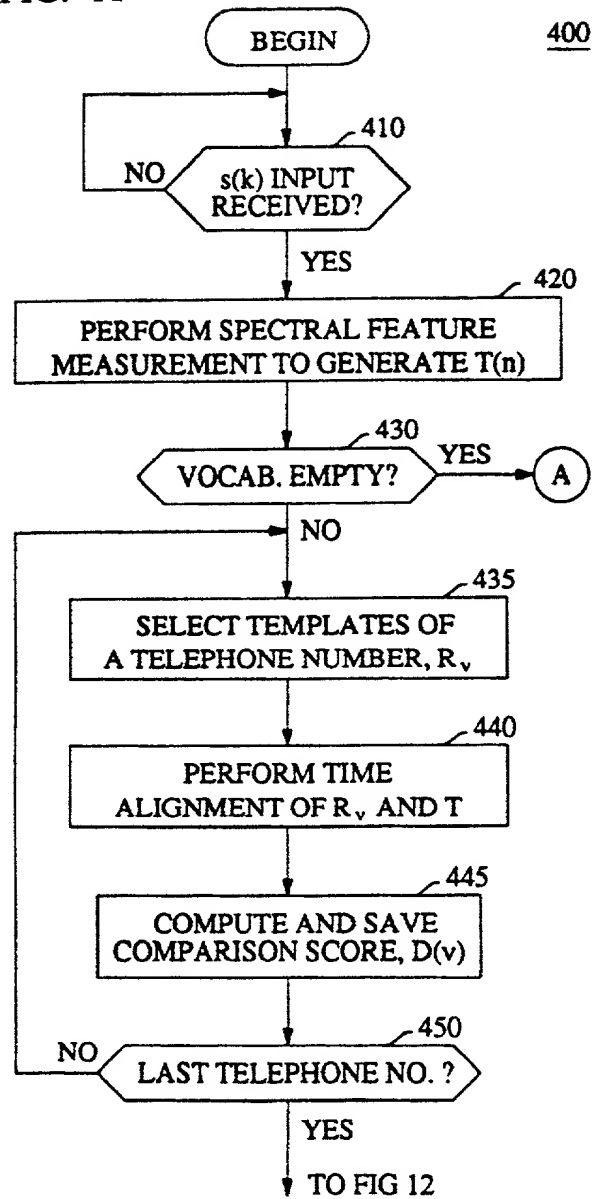
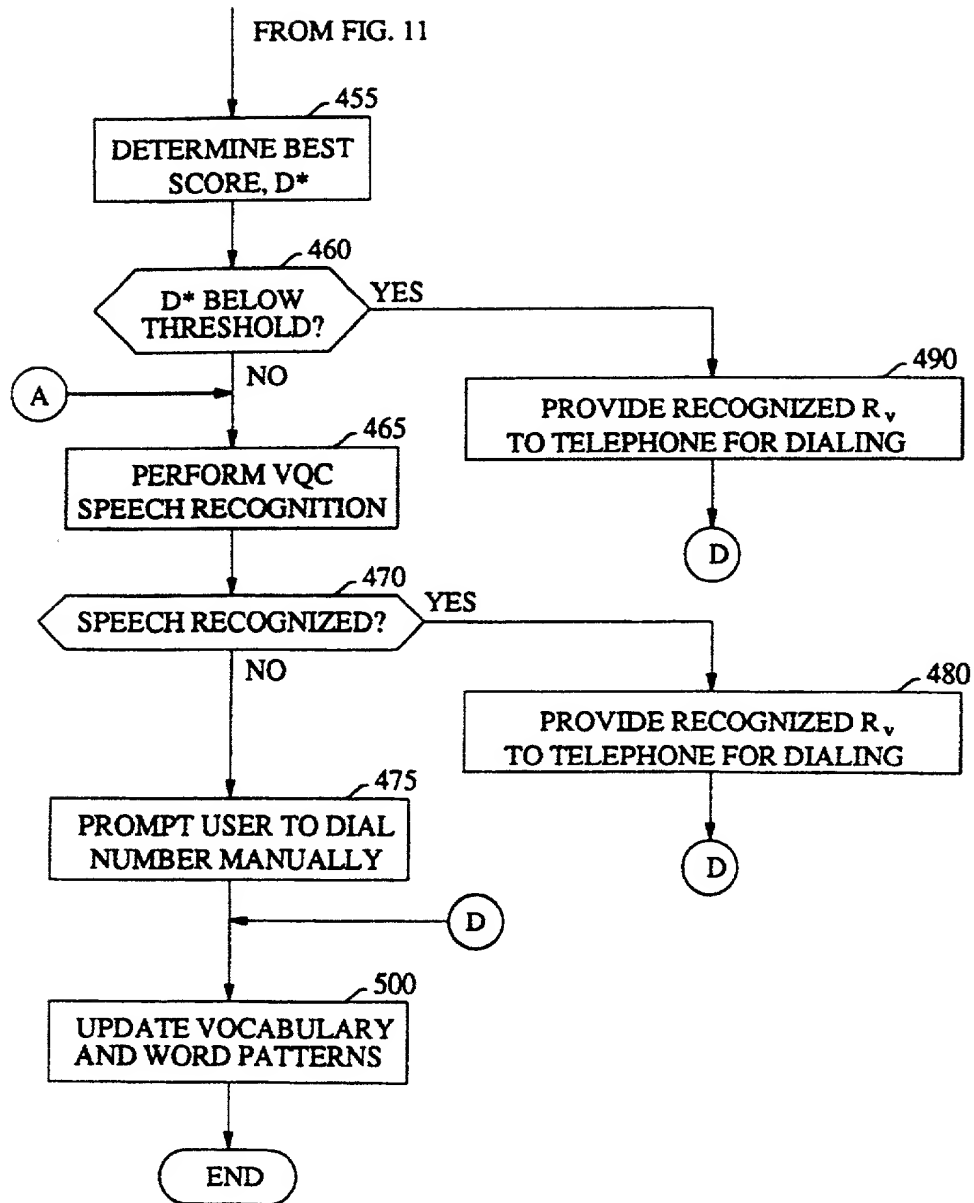


FIG. 12



Requested Patent: JP6204952A

Title:

METHOD FOR TRAINING VOICE RECOGNITION SYSTEM FOR TELEPHONE LINE UTILIZATION ;

Abstracted Patent: JP6204952 ;

Publication Date: 1994-07-22 ;

Inventor(s): STANFORD VINCE M; BRICKMAN NORMAN F ;

Applicant(s): INTERNATL BUSINESS MACH CORP It;IBMgt ;

Application Number: JP19930219208 19930812 ;

Priority Number(s): ;

IPC Classification: H04B14/04 ; G06F3/16 ; G10L5/06 ;

Equivalents: JP2524472B2

ABSTRACT:

PURPOSE: To provide a system with which continuous voices inputted from a telephone line can be recognized by inputting a correction voice data set to a voice recognizing processor and training a statistical pattern matching unit.

CONSTITUTION: A digital voice input from an A/D 100 is sampled by a data rate converting device 102, a voice data stream is divided into fine frames by a vector-quantizing block 104, and characteristics are extracted from the respective frames and summerized. Vector quantization is similarly used in a training process 190, as well. A quantizing code book 105 preserves a code book for telephone voice generated by the function 190. On the other hand, input voice data 180 are resampled in a block 182, supplied to a code/decode band filter and set to a prescribed band width. The correction voice data set is inputted to the voice-recognizing processor, and the statistical pattern matching unit is trained. The voice recognition is executed, while using a voice recognition processor from a telephone system.



09/930395

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平6-204952

(43)公開日 平成6年(1994)7月22日

(51)Int.Cl. ⁵	識別記号	庁内整理番号	F I	技術表示箇所
H 0 4 B 14/04		Z 4101-5K		
G 0 6 F 3/16	3 1 0	A 7165-5B		
G 1 0 L 5/06		Z 9379-5H		

審査請求 有 請求項の数7(全 13 頁)

(21)出願番号 特願平5-219208

(22)出願日 平成5年(1993)8月12日

(31)優先権主張番号 07/948, 031

(32)優先日 1992年9月21日

(33)優先権主張国 米国 (U S)

(71)出願人 390009531

インターナショナル・ビジネス・マシー
ズ・コーポレーション

INTERNATIONAL BUSIN
ESS MASCHINES CORPO
RATION

アメリカ合衆国10504、ニューヨーク州

アーモンク (番地なし)

(72)発明者 ビンス エム スタンフォード

アメリカ合衆国メリーランド州ゲイサスバ
ーグ フラコンブリッジ テラス 15349

番地

(74)代理人 弁理士 合田 深 (外2名)

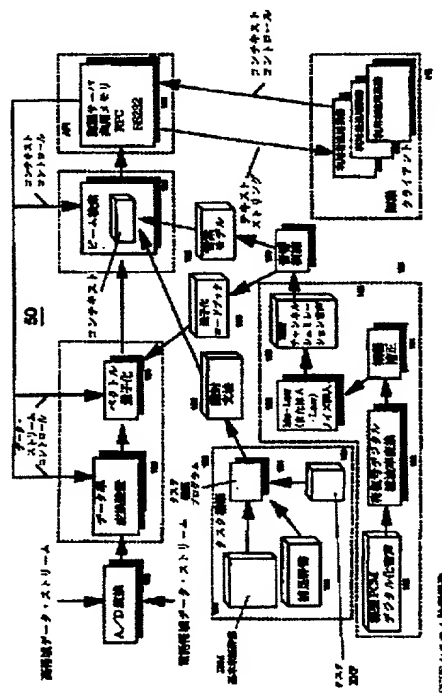
最終頁に続く

(54)【発明の名称】 電話回線利用の音声認識システムを訓練する方法

(57)【要約】

【目的】 本発明は、電話回線から入力される連続音声
を音声認識できるシステムを提供する。

【構成】 本発明は、電話システムからの音声に対応で
きる音声認識システムを訓練するための電話チャネル・
シミュレーションを開示する。電話音声帯域より高い帯
域を持つ音声認識訓練プロセッサに音声データ・セット
が入力される。入力音声データ・セットを修正する一連
の処理が行われ、最終的に修正される音声データ・セッ
トによって音声認識プロセッサが電話からの音声信号の
音声認識を行うことが可能となる。



1

【特許請求の範囲】

【請求項1】 電話帯域幅より高い帯域幅の音声認識訓練プロセッサへの音声データ・セットの入力ステップと、

上記音声データ・セットを間引き、上記電話帯域幅を有する間引かれた音声データ・セットを入手するステップと、

帯域通過デジタル濾波器を上記間引かれた音声データ・セットに適用し、電話機器の電送特性に特性化した、濾波された音声データ・セットを入手するステップと、

上記濾波された音声データ・セットの振幅を、その最大ダイナミック・レンジが非圧伸電話音声の最大ダイナミック・レンジと一致するように補正し、振幅補正音声データ・セットを入手するステップと、

上記振幅補正音声データ・セットを、電話システムの圧伸・非圧伸音声信号シークエンスを表す量子化ノイズを用いて修正し、修正音声データ・セットを入手するステップと、

上記修正音声データ・セットを音声認識プロセッサに入力し、統計的パターン・マッチング・ユニットを訓練するステップと、

から構成される、電話システムから得られる音声にตอบสนองする音声認識プロセッサを訓練する方法。

【請求項2】 上記電話帯域幅が上記音声幅の高位帯域より低い帯域である上記請求項1記載の方法。

【請求項3】 上記帯域通過デジタル濾波器が最大平坦設計アルゴリズムを備え持つ上記請求項1記載の方法。

【請求項4】 上記音声データ・セット振幅補正の結果、最大ダイナミック・レンジが非圧伸 μ -law 電話音声の最大ダイナミック・レンジに一致する上記請求項1の記載方法。

【請求項5】 上記音声データ・セット振幅補正の結果、最大ダイナミック・レンジが非圧伸 A-law 電話音声の最大ダイナミック・レンジに一致する上記請求項1記載の方法。

【請求項6】 上記音声データ・セット修正ステップが μ -law ノイズとしての量子化ノイズを用いる上記請求項1記載の方法。

【請求項7】 上記音声データ・セット修正ステップが A-law ノイズとしての量子化ノイズを用いる上記請求項1記載の方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、公衆電話交換回線を利用する音声認識システムに関するものである。

【0002】

【従来の技術】 音声認識システムは、よく知られている技術である。IBM タンゴラ (Tangora) [13] (本願書文末記載の参考文献の番号で、以下同様に表記する) およびドラゴン・システム・ドラゴン 30 k 口述システムは

2

その例である。それらは、典型的な単一ユーザおよび話し手依存型システムである。これは、プロセスが「登録」と呼ばれるプロセスの間に、話し手の音声パターンで音声認識装置を訓練することを各話し手に要求する。将来の認識セッションの中で話し手自身をシステムが識別しなければならないのでシステムは話し手のプロフィールを維持する。典型的には、話し手は低レベル雑音システム環境の中でローカル・マイクを通して認識システムが常駐する単一の機械に話しかけながら登録を行う。登録作業の間、その話し手は、長ったらしい原稿を読むことを要求されるが、それ故に、そのシステムは各話し手の特色に順応することができることとなる。独立した口述システム (たとえば、上記の2つのシステム) は、話し手にたどらどしい、不自然な形で、すなわち、語と語の間にポーズをいれながら、各語を形づくることを要求する。これにより、音声認識システムは、語の境界となる、先行および後続の無音を利用し、各個人の語に連想される音声パターンを識別することが可能となる。典型的音声認識システムは、(たとえば、IBM タンゴラ システムの Office Correspondence の場合のように) 単一の機械上で作動し、訓練された単一の適用業務を持つ。

【0003】 話し手依存型音声認識装置をもつマルチ・ユーザ・システム環境は、各話し手にその音声パターンをシステムに理解させるための退屈な訓練に従事することを要求する。話し手の電話番号によってシステムがどの音声テンプレートを使用すべきかを知り得る共通データ・ベースに音声テンプレートが格納されているかもしれないが、それでもなお各話し手は使用の前にそのシステムを訓練しなければならない。外部の電話線から接続してくる新しいシステム利用者は、このプロシージャが容認できるものでないことを認識する。また、成功した電話の音声認識システムというものは、様々の分野に関係する音声を正確に認識するために迅速な文脈切り替えができなければならない。たとえば、一般のオフィス通信のために訓練されたシステムは、数字列の提示の場合、うまく働かない。

【0004】 Kai-Fu Lee の博士号論文[1]の中で最初に記述されたスフィンクス (Sphinx) システムは、以前の話し手依存型認識システムに大きな進歩をもたらした。それは話し手独立型であり、会話音声の連続ストリームから単語を認識することができた。このシステムは、実際の使用に先立って行われる話し手個々の登録を必要としなかった。話し手依存型システムの中には、話し手に4〜6週毎に再登録することを要求したり、利用者にそのシステムが理解するための個人用プラグイン・カートリッジを持ち運ぶことを要求する。連続の音声認識を行うスフィンクス・システムは、語と語の間の休止を必要とせず、音声認識システムの一時的ユーザに非常に多くの親切的なアプローチを提供する。認識システムの利便のた

【0005】音声認識システムは、また、与えられたやさやかな語彙を使って、実時間処理を提供しなければならない。しかし、スフィンクス・システムは、まだ以前の話し手に依存する認識システムの不利な点をいくつか持っていた。マイクロホンおよび比較的に制約された語彙を使用しながら低レベル雑音システム環境の中で単一機械上で操作するようプログラムされていた。スフィンクス・システムは、複数ユーザのサポート、少なくとも、異なるロケーションおよび複数の語彙認識に関するサポートを行うようには設計されなかった。

【発明が解決しようとしている課題】本発明は、上記の従前技術の不利益な点の多くを克服することを目的とする。したがって、本発明は、ローカルおよび遠隔地双方の話手からの入力を持つ電話機器使用に適した連続音声話し手独立型音声認識システムを提供することを目的とする。

【0008】複数の音声適用業務が、コンピュータ・ネットワーク上または電話線上で同時に音声認識システムによって音声認識されるようにすることは、本発明のもうひとつの目的である。

【課題を解決するための手段】本発明の上記目的は、ローカル・エリア・ネットワークまたは広域ネットワークの上のクライアント・サーバを基に構築される音声認識システムによって達成される。この音声認識システムは、アナログまたはデジタル音声データを音声を表わす一組のケプストラム係数およびベクトル量子化値に変換するフロントエンドを含む多くのモジュールに分けられる。バックエンドは、ベクトル量子化値を使用して、その音声の作る文脈と音素モデル(Phoneme Models)と語対文法(Word Pair Grammars)に従ってその語を認識する。語彙を一連の文脈(すなわち、ある特定の語がそのシステムによって予期される状況)に分割することによって、一層大規模な語彙を、最小限のメモリに収納することができる。ユーザが音声認識作業を進めるにつれて、文脈は共通のデータベースから迅速に切り換えられる(下記引用Brickmanその他による特許出願参照)。システムは、また、コンピュータ・ネットワーク間および複数のユーザ適用業務間のインターフェースを備える。

10

20

30

40

5

【0012】

【0013】データ収集：データは、アナログからデジタル形式にブロック100で変換されるか、あるいは電話のデータの場合他のチャネルから潜在的にデマルチプレックス(demultiplexed)される。

【0014】データ圧縮：ICSRフロントエンドブロック102および104は、ベクトル量子化ステップの間に300バイト/秒に音声データストリームを調整し、再標本化し、圧縮する。

【0015】音声認識：バックエンド106は、文法ガイド型ビーム・サーチ・アルゴリズムを使用しているパターンマッチング音素モデル192によって実際の音声

5

認識を実行する。音素モデル192および語対文法135は共に認識文脈を構成する。バックエンド認識装置のひとつまたは複数の事例が、遠隔地であろうがローカルであろうが音声データを捕捉し圧縮するフロントエンド事例に配備されることができる。

【0016】タスク構築：タスク構築コンポーネント130は、認識文脈のオフラインでの構築を可能にし、実行時で使用のために語対文法をコンパイルし、適切な音素モデルをそのタスク（文脈）に連結させる。

【0017】適用業務プログラム・インタフェース（API）：API108は、データストリーム・コントロール、文脈ローディングおよび起動を可能にするRPC（Remote Procedure Call）に基づく認識サービスを提供する。

【0018】電話チャネル・シミュレータ：シミュレータ185は、高帯域、高解像度音声データ・セットを、音素モデル192および電話音声に連結し、減少された標本抽出率、圧縮された帯域幅および圧縮されたダイナミック・レンジの電話音声を作り出す。

【0019】音声認識の間に、ローカル・マイクからの高帯域音声データストリームも電話に関連しているような低帯域音声データストリームも、アナログデジタル変換ブロック100によって受け取られる。アナログデジタル変換100は、ボイス・ワークステーション上のIBM M-Audio Capture/Playback Cardカード（M-ACPA）のようなハードウェア・カードによって実行されることができる。M-ACPAは、高帯域または電話帯域幅信号を処理するデジタル信号処理機構を持ち、デジタルに標本化された一連のデータ・ポイントにそれらを変換する。この変換は、また、デジタルPBXや8KHz、8ビットのMu-Law/A-Law形式で与えられる電話データストリームによって実行されることもできる。

【0020】本発明では、高帯域を、サンプル率16キロヘルツ以上と定義する。低レベル帯域幅を、アメリカ合衆国で一般の電話がデジタル音声に使う8キロヘルツ以下と定義する。電話システムの中でデジタル情報が個人の電話交換（PBX）から入る可能性があるため、A/D変換ブロック100は、オプションとして必要である。

【0021】音声認識に対する「フロントエンド」の中の最初の重要なブロックは、データ条件付け・速度変換ブロック102（Data Conditioning and Rate Conversion）のことで、以下DCRCと呼ぶ）である。A/D変換100からのデジタル化された入力、44または8KHzである。本発明で間引き（DECIMATION）と呼び使用する再標本化テクニックは、IEEEの文献(2)によって提供されている。DCRC102は、デジタル化された信号に対しアンチエイリアシング（Anti-aliasing）・フィルターを使用し標本化を行い、次のステップでの使

6

用のために、16KHzまたは8KHzデータストリームを作る。DCRCおよびベクトル量子化プロセスは、以下に詳細に記述される。

【0022】音声認識の中でデータ条件付け・速度変換の後、音声データは、ベクトル量子化ブロック104に渡される。ベクトル量子化の中でデジタル・データ・ストリームは、1秒間の1/50のフレームに細分化され、16KHz、11KHzおよび8KHzそれぞれの標本化率に対し各々320個、220個および160個の標本となる。本発明の好ましい実施例のひとつでは、いかなる帯域幅音声信号からも計算される1秒につき100フレームがあり、それらは50パーセント上重ねられ、ハミング・ウィンドウ（Hamming Window）が適用される。ハミング・ウィンドウは、文献(3)で定義されている。

【0023】音声データストリームがフレームに細分化されたあと、ベクトル量子化ステップは、各フレームから特性を抽出する。ベクトル量子化ステップの抽出部分で、LPCケプストラム係数と呼ばれる一連のパラメータが、計算される。ケプストラム係数は、パターン認識のために音声の重要な特性のいくつかを抜き出し、要約する。データの各フレームの中で、音声の1秒の50分の1が、カプセルに入れられる。1秒につき50のフレームと想定するであろうが、50パーセントの上重ねがあるので、1秒につき100フレームが生成される。ケプストラム係数を計算するために、まず（コサイン・ベル-cosine bell-である）ハミング・ウィンドウが、音声データに適用される。抽出されたデータが、無限時間連続フーリエ変換にあるようにするために、ハミング・ウィンドウは、音声データの各フレームのエッジを次第に減少させる。

【0024】ハミング・ウィンドウ化されたフレームは音声スペクトルを平坦にするために、そのZ変換が $1.0 - 0.97 * z^{-1}$ （[1]49ページ参照）であるところの濾波器を使用して事前に濾波される。それから、14個の自己相関係数が計算される。自己相関係数が、文献(4)の記述でよく知られている方法でケプストラム係数を計算するために使われる。13個のケプストラム係数は、14個の自己相関係数から引き出される。自己相関係数の数やケプストラム係数の次元数を変えることは可能である。これらの係数の統計的特性は、最終的なベクトル量子化ステップをガイドするために使われる。

【0025】ベクトル量子化は、訓練プロセス190の中でも同様に使われる。下記の訓練データの調整は、基本スフィンクス認識エンジンを電話機器上で作動可能とさせる点で、本発明にとって重要である。訓練プロセス190において、10,000から15,000の間のセンテンスがとられて、フレームに細分化され、そこから自己相関およびケプストラム係数が計算される。参照文献(5)に記述されるk-手法タイプのクラスタリング

・プロシーダを使用して、256個のクラスにケアストラム・フレーム特性を区分する。これらのケアストラム・クラスターの中央値、およびそのクラス・ラベルが共に取り出され、これ以後「コード・ブック」と呼ばれる。量子化コード・ブック105は、音響訓練機能190によって生成される電話音声用コード・ブックを保存し、また、第2の高帯域音声用コード・ブックをも保存する。

【0026】ベクトル量子化の最終的なステップのために、どのクラスター中央値がフレーム・ケアストラム係数に最も近いかを決定するために、ブロック104は、上記のように訓練プロシーダで引き出される量子化コード・ブック105のコード・ブックを参照する。現在のフレームが、コード・ブック値によって表わされたクラスに割り当てられる。256個のクラスがあるので、VQ(Vector Quantization)値は、1バイトで表わされる。微分ケアストラムおよびフレームのそのべき乗から引き出される別の2個の1バイトVQ値がある。1秒に100回引き出される3個の1バイトVQ値があり、その結果、音声データストリームは2、400ビット／秒に圧縮される。

【0027】音声認識装置のためにその音声の特徴づけるところの完全に別個のコード・ブックが、電話データから引き出され、図1の量子化コード・ブック105で保存されなければならないということは、電話音声認識に関する本発明の一部である。また、対応する音素モデルが電話データから引き出され、音素モデル192で保存されなければならないということは、本発明のもう一つの部分である。標本率減少、帯域幅圧縮およびダイナミック・レンジ圧縮のために、電話音声信号はかなり変わる。しかし、多大な労力を要する、電話から収集する音声標本の使用を必要とせず、高帯域標本を、電話チャネル特性をシミュレートするように処理することができる。これにより、スフィンクス・システムの初期化訓練で使われた、大規模で既に使用可能な音声データ・ファイルを活用して、電話音声認識を可能となる。電話チャネル・シミュレータは、本発明の対象である。

【0028】電話チャネル・シミュレーションは、下記の通り、3つの段階的プロセスで達成される。

1.) 電話帯域幅への変換

文献[14]から[19]で参照されるように、(たとえば44、100Hz、あるいは16、000Hzで集められた16ビット解像度データのような)高帯域、高解像度音声データ・セットが図1のブロック180への入力となる。

【0029】入力音声データ・セット180は、最初に、図1のブロック182の中で[2]で記述の再標本化プログラムを使用して、8、000Hzに再標本化される。このデータは、図1の機能ブロック182で、参考文献[8]で記述のMAXFLATルーチンの修正版を

使用して設計された符復号器帯域濾波器に供給される。この濾波器は、図2、3および4の中で図示される。この濾波器の通過帯域特性は、現代の米国における電話機器の中で使われる符号化/復号化濾波器に近似するように設計される。通過帯域、3dbポイントおよび移行(TRANSITION)帯域幅の設定は、本発明の有効性にとって重要である。ローカル電話回線上の音声に対する良好な認識を行う認識訓練のための符復号濾波器を設計するのは可能であるが、遠隔地の電話については難しい。そのような問題を避けるために、上記の特性は、たとえば、低位の3dbポイントに対しては300Hz、上位の3dbポイントに対しては3、600Hzに設定すべきである。移行帯域幅は、それぞれ、400Hzおよび800Hzでなければならない。通過帯域は500Hzから3、200Hzになる。実際の符復号器濾波器の幅に近似するために、通過帯域リップルは、全通過帯域にわたり、1単位から0.1パーセント以上の偏差であってはならない。

【0030】スフィンクス音声認識エンジンおよびタンゴラを始めとするその他の音声認識エンジンが線形濾波器によって提示されるスペクトルのひずみを感じ取れる点に、注意することは重要である。スペクトルのひずみは、主要な音声認識特性(例えばケアストラム)が周波数スペクトルから引き出されるので、その通過帯域の中の平坦な周波数応答を持たない。複雑な認識作業については、いくぶん平坦な通過帯域応答からのマイナーな偏差が、本願発明者の研究室において観察され、結果として、絶対認識誤り率が数パーセント劣化した。したがって、最大平坦設計アルゴリズムは、必要である。「スペクトルの傾き」へのスフィンクス音声認識エンジンの感度が、参考文献[9]の中で指摘された。したがって、MAXFLATまたは比較的低レベル通過帯域のリップル設計は、必要とされる。

【0031】4、100Hzから8、000Hzへの再標本化率変換は、参考文献[8]の中で提供されたMAXFLATには過度な要求であり、それは、帯域通過特性が符復号器帯域濾波器に必要なとき、低通過帯域フィルターの設計のためにのみ役立てられる。このルーチンに対するデザイン特性は、0.5へマップするナイキスト周波数と1.0へマップする標本化周波数によって、正規化された周波数の3dbポイントおよび移行帯域幅を表わす2個のパラメータ、ベータおよびガンマによって与えられる。Kaiserの参考文献[8]によって、ガンマは「0.005よりあまり小さくない」値に制限されなければならないことが示唆されている。これより低い値では、使われる計算精度浮動小数点数を増やすためにルーチンの修正が要求であり、そのような濾波器の条件数は、およそガンマの2乗に反比例するので、フィルター係数バッファを200から4096に拡張する必要がある。このため、44、100Hzから8、000Hzへの交換に必要となる0.05の約10分の1または0.0

0.5のガンマ値をもつ濾波器とした。2個の低域濾波器設計、低域から高域通過帯域変換、および、低域と高域通過波の渦状組合せが、必要な帯域通過特性を実現するために要求された。

【0032】上記フィルター設計の実現によって、4
4、100Hzデータは、参照文献[2]で記述される再
標本アルゴリズムを使用して、図1の機能ブロック18
2の中で8、000Hzに変換され、米国長距離電話機
器のための通過帯域に非常に近い符号器通過帯域を提
供する。このデータは、下記のステップ2および3に従
って処理され、16ビットの、低雑音信号となる。

【0033】同様の通過帯域特性および速度低減削減は、この訓練テクニックの中で使われる16,000Hz音声サンプルのために必要であるが、例外は、移行バンド要求がそれほど要求していない点と濾波器加重が、要求された通過帯域平坦度特性を達成するにはさほど必要とされない点とである。図2、3、4で、事前訓練再標準化操作の訓練に実行されたのと同様に、符号器濾波器のインパルス(Impulse)、マグニチュード(Magnitude)およびログ・マグニチュード(Log Magnitude)応答を再びを示す。

2) ダイナミック・レンジを正規化するための振幅補正 (スケーリング)

音声標本は、個別に読まれて、図1のブロック184で、14ビットのダイナミック・レンジにスケーリングされる。

3) Mu-law 压伸

各音声標本は、図1のブロック186で、参照文献[7]のような公の文献でよく知られているMu-law圧縮を使用して16ビットの精度から8ビットの精度に引き下げられる。8ビットへ圧縮されたデータは、ふたたびMu-law公式に従って、14ビットへ拡大される。

【0034】この結果、図1ブロック188でシミュレートされる電話チャネル音声データ・セットになる。これは、信号強度によって増大、減少する量子化ノイズ・レベルを持ち、およそ一定のS/N比を維持する。特に、話し手の声の大きい場合、これは、電話音声信号の中で聞きとれる「ひび割れ」雑音を導入する。

【0035】電話データより高域の種々の帯域幅で集められるであろう音声データ180のこのような処理は、電話機器での使用のため音声認識装置50をブロック190で訓練するために使用される。音響訓練190は、図1のブロック192の音素モデルと量子化コード・ブック105を生成する。これにより、スフィンクス音声認識エンジンを使用して電話帯域幅での実際の音声認識を行うことを可能とする。

【0036】シミュレートされた電話チャネルデータ使用の認識装置訓練

2個のコード・ブック105と2個の音素モデル・セット192が作成されるように、2つの訓練セッション、

すなわち電話と高帯域に対するセッションが、実行される。高帯域、ローカルな認識あるいは、電話帯域幅などのユーザの要求に応じてコード・ブック105の各セットおよび各音素モデル192は、別々にに保管され、実行される。いずれの帯域幅でも、自己相関係数は、ケプストラム係数を引き出すために抜き出される。そのフレームにもっとも近い係数を類別するために、ケプストラム係数がベクトル量子化104によって実行される。このようにして、{1}で記述されるように、各音声時系列フレームは、そのフレームを表わす3バイトに減じられる。

【0037】量子化の値のセットが、ビーム・サーチ・プロセス106に送り出される。ビーム・サーチ106は、ビタービ(Viterbi)ビーム・サーチと呼ばれる文法ガイド型「隠れたマルコフ・モデル」(Hidden Markov Model)サーチ・プロセスである。この文法ガイド型サーチは、サーチ・スペースを減らすために語対文法を使う。

【0038】本発明のもうひとつの重要な点は、その音声認識システムがローカルであろうが遠隔地であろうが、両方の音声処理することができることである。これは、音声のいずれのタイプもチャネル・シミュレータで使われる帯域幅に対応するように、実行時データ条件づけ・速度変換滤波器の遮断ポイントを2個の帯域幅の幅が狭い方に近い帯域幅に置くことによって、達成される。3dbポイントおよび移行帯域特性は、訓練の中で使われる電話符復号器滤波器の上位移行帯域の特性に近似しなければならない。

【0039】ビーム・サーチ106は、そのベクトル量子化の中で引き出された時系列を語対文法からの語列に突き合わせ、各文脈を定義する。音声認識サーバは、ユーザ適用業務または音声認識クライアント（ブロック110）とコミュニケーションする。本発明の構造は、単数のバックエンドとコミュニケーションする複数のフロントエンド（ワークステーション）または複数のバックエンドとコミュニケーションする複数のフロントエンドを持つことができる。

【0040】本発明のシステムは、オペレーションの異なるレベルのために構成され実行される。非常に高いデータ速度をもつコミュニケーション・ネットワークについては、フロントエンドでのデータ圧縮のために、音声標本は、直接バックエンドを実行しているシステムに伝達されることができる。原デジタル音声データストリームが、複数のユーザ用のバックエンドがあるサーバに送り出されることができる。電話システムについては、複数のチャネルが1つのバックエンドへつながるか、または、複数のユーザが、フロントエンドおよびバックエンド双方にコミュニケーションする。

【0041】本発明でのシステムは、音声認識サーバとして配備される音声認識機能を中心に主として構成さ

れる。システムは、その時点の文脈として適用業務が選択する語対文法によってガイドされる。音声認識適用業務は、初期値設定プロシージャ、ステータス・コードおよびコマンド〔6〕のような機能をサポートする適用業務プログラム・インタフェース(API)コールをもつ音声認識システムにインタフェースを持つ。音声認識適用業務は、音声認識サーバに一定のタイプの操作を要求するか、あるいは、ある特定の認識文脈をロードして、必要なとき、音声認識のための文脈を起動するよう要求する。音声認識適用業務が最初に行われるとき、タスクは通常サーバによって事前ロードされる。適用業務の活動の必要に応じて、タスクはその後順に起動される。

【0042】音声認識サーバ(ブロック108)のAPIコールは、ユーザ適用業務(ブロック110)が音声認識システムのサービスを要請することを可能にする。ユーザ適用業務プログラム(ブロック100)は、音声認識サーバの種々の構成要素と同じコンピューターまたは異なるコンピューターの上で実行することができる。同じコンピューター上の場合、適用業務プログラム(ブロック110)は、そのオペレーティングシステムでサポートされる共有メモリおよびセマフォを通して音声認識サーバとインタフェースをとることができる。異なるコンピューター上の場合、通信はRS232Cインタフェースあるいは遠隔プロシージャ呼出し(RPC)を通して行われる。RPCは参照プログラミング文献〔10〕でよく知られている。

【0043】ユーザ適用業務の典型的例には、エグゼクティブ情報システム、言葉の照会経由のデータベース・アクセス、ソフトウェア問題報告システムなどがある。

【0044】もうひとつの例は、その利点を活用するため音声認識サーバへの呼び出しを行う電話回答音声応答装置(VRU)である。RISC SYSTEM 6000(TM)およびOS/2(TM)をもつPS/2(TM)の上でこれらのサーバは実行された。

【0045】Direct Talk 6000(TM)は、同様の電話VRUシステムである。このVRUシステムでは、1本の電話回線を扱うのではなく、(同時に活動中となる可能性のある24個の会話チャネルをもつ)T1回線処理が必要となる。音声認識サーバ構造は、Direct Talk(TM)のように大量の電話適用業務の処理が必要な場合、複数のクライアントを扱うことができる。ユーザ適用業務は多くの文脈を前もって登録することができる。レストラン案内、ハードディスク・ヘルプ・デスク、あるいは、ソフトウェア・ヘルプ・デスクは全て複数の文脈を階層的に事前に登録することができる。各適用業務では、何人かのユーザが、音声ストリームを入力することができる。各適用業務は、特特有の音声ストリームのために特有の文脈の下で音声認識を実行するよう音声認識サーバに指示する。

【0046】言い換えると、同じAPIを扱う複数のユ

ーザが、1またはおそらくいくつかの版の音声認識サーバを用いるタスクすべてを登録するであろう。システムは、要請された作業がすでにロードされているかを検査し、複数のユーザの音声認識タスクが余分にロードされることを回避する。

【0047】タスク構築(ブロック130)は、いくつかの基本力ソースを持つ。20,000語の発音をもつ基本辞書である米語辞書(ブロック132)は、その1つである。補足辞書(ブロック138)は、適用業務特有のもので、基本辞書の中で見つけれなかった語の発音を追加するためのものである。補足辞書は、典型的には、特定の適用業務が音声認識のために必要とする固有名詞、頭字語(ACRONYM)その他から構成される。

【0048】基本米語辞書(ブロック132)は、タスク構築プログラム(ブロック134)によって求められる語および音素を供給する。タスク構築プログラムは、また、何かがそのタスクの下で音声認識サーバによって認識されることができるかを決めるためにタスクBNF辞書(ブロック136)から該当するタスクBaukus-Naur Form(BNF)文法を引き出す。たとえば、地域レストラン情報を提供する適用業務の最初の文脈は、その話し手が希望するレストランのタイプ、たとえば、フランス、イタリア、中国料理などであるかもしれない。ひとたびそのタイプが決まれば、次の文脈は、その特定のカテゴリの中のレストランとなろう。タスク構築プログラムは、そのパターン合わせのために必要なすべての語を見つけるためにBNFを分析し、汎用の米語辞書(ブロック132)から音素表示を引き出す。必然的に、あらゆる特定適用業務は、そのシステムに加えられるなければならないそれ自身の副語彙を持ち、それらは、補足辞書に保存される。たとえば、レストラン・ヘルプ・デスクの中で、「イタリアン」、「フレンチ」、「スパニッシュ」などの言葉は、汎用米語辞書で見つけられるが、レストラン名、とくに外国語で、たとえば、「Cherchez Les Femmes」、「Chateau Voulez」や、アメリカのレストランで普通でない名、たとえば、J. J. Muldoon、は、普通の辞書になく、タスク補足辞書(ブロック138)に加えられる。これらの補足辞書(ブロック138)は、また、基本汎用米語にあるが発音をローカルなものにするためにローカルな語彙を含めることができる。

【0049】タスク構築プログラム(ブロック134)は、入力BNF文法を分析して、その文法の中の各語のリストと次に続くことができるすべての語のサブリストを生成する。したがって、その文法の中の各語が、後に続く適切な語のリストおよび各語の音素表示へのポイントを持つ。音素モデル192は、種々のVQ値を観察するである。このマルコフ・モデルは、VQ値(ブロック104)のための、一群の離散的確率分布であり、「隠れたマルコフ」状態機械が音素の範囲内の特定の状態に

あるとすると、VQ値のオカレンスの確率を与える。
「隠れたマルコフ・モデル」は文献[11]に適切に記述されている。

【0050】ビーム・サーチ(ブロック106)は、訓練プロセスの間に生成される文脈感知のトリフォン(tri-phones)の大きいテーブルから連結HMM音素モデル192でできている語モデルを使用する。この語モデルが、VQ値の観察された順序を最もよく説明する語順序の最適推定を行うために使われる。ビーム・サーチ(ブロック106)は、そのサーチの中で使われる語をつくるための音素モデル192を選択するために、語文法を使う。

【0051】ユーザ適用業務は、音声認識サーバを制御する。例えば、[12]で記述されるIBMプログラム・プロダクト Direct Talk/2(TM)は、電話に応答しレストラン案内機能を実行するひとつのユーザ適用業務となり得る。レストラン案内適用業務は、Direct Talk/2(TM)を使用し、この適用業務が16の文脈を持ち、レストラン案内ヘルプ・デスクの一部である文脈を事前ロードする要求を起こすことを音声認識サーバに知らせる。その適用業務が進行するにつれて、音声認識サーバの文脈切り替えを要請する。ユーザは、電話ヘルプを電話を通して呼び出す。レストラン案内は、音声認識サーバに最初のレベルの文脈での音声認識を実行することを要請する。認識サーバとユーザ適用業務間のAPI上で制御とデータが交換される。Direct Talk/2(TM)システムの複数の事例が同じ音声認識サーバを使用する可能性がある。

【0052】音声認識サーバは、無声間隔(ユーザが調整可能で、ほとんど一般に0.6秒)が来るまで音声データを捕捉する。無声間隔が観察されると、認識は終了し、話し手の話しが終わったと仮定される。

【0053】本発明記載の音声認識システムは、複数のハードウェア・プラットフォームおよび複数のソフトウェア機械構成の上に、複数の実施を可能にするよう基本設計がなされる。たとえば、1つの可能な構造は、図5のように、ローカル・エリア・ネットワーク160を通して接続されているワークステーションの物理的实施の上への上記論理的構造50の物理マッピングを提供する。この構造の中の各ワークステーション150、150'、150''は、複数の独立ユーザ適用業務を実行することができ、各々は、スレーブ・プロセッサとしての音声認識サーバ50のマスターとなる。PBX170は、外部の電話回線に接続していて、電話帯域幅データストリームを図1の音声認識サーバ50のアナログ・デジタル変換100に渡す。認識された音声を表わしているテキストが、ワークステーション150、150'、150''のユーザ適用業務に音声認識サーバから返される。

【0054】訓練プロセス

ビーム・サーチ・プロセスでの語モデルとテキストとのパターン合わせに使われる音素HMM192のパラメータを推定するために、訓練プロセスは、既知の音声およびテキスト原稿という大規模辞書を使用する。

【0055】最初に、その原稿が、訓練セットの語の発音を表わす音素を汎用米語辞書から検索するために使われる。

【0056】次に、音素HMM192のパラメータが、共調音(coarticulation)効果の効果的な推定を行うために、先行および後続音素文脈(トリフォン-triphonesと呼ばれる)の中で推定される。使われる推定プロセスは、[11]で記述のBaum-Welch 順方向、逆方向繰返しアルゴリズムである。訓練されたトリフォンHMMが訓練セットの中で観察されたVQ値時系列を生成したであろう確率を最大にするために、HMMのパラメータが、繰返し調節される。

【0057】あらゆる「隠れたマルコフ」音素モデルには多くのパラメータがあり、各「隠れた」状態機械中に7つの状態および12のトランジション・アーク(TRANSITION ARC)が存在する。各トランジション・アークに関連して、3つのコード・ブックの各々の確率分布に、関連する256の離散エレメントがある。訓練プロセスから生じるトリフォンHMMパラメータは、連続音声の中に存在する共調音効果を表わすのに必要なトリフォン数を減らすために一定の幅の値の範囲に集められる。

【0058】訓練は、ローカルな電話交換を通して集められる低レベル帯域幅音声およびマイクからの高帯域音声の組合せによって実行される。高帯域音声は、本発明に従って、本願書で記述の電話チャネル・シミュレータ185によって処理される。3つのコード・ブックすべては、この段階でコンパイルされる。[1]で記述のように、コード・ブックが、ケプストラム、微分のケプストラム、べきおよび微分のべきを含む。

【0059】3つのコード・ブックの各々は量子化コード・ブック105に保存され、実行時ベクトル量子化プロセスで使われる。ここで、電話ネットワークの効果、データの事前処理によってシミュレートされ、公衆電話ネットワークが調整するのと同じ方法で特性コード・ブックの統計的屬性が調整される。このプロセスをとることによって、米国大陸の様々なロケーションからの呼び出しをもつ実際の電話の音声認識の正確度が大幅に増加した。

【0060】図6は、たとえばPBX170経由で電話から得られた音声に回答する音声認識装置50を訓練するための電話チャネル・シミュレーション・プロセス200を記述する流れ図である。図6の流れ図は、図5のデータ処理装置50の上で実行されることができるコンピュータ・プログラム方法を表わす。

【0061】プロセス200は、電話帯域幅より帯域幅がより高い音声認識訓練プロセッサ50に音声データ・

セットを入力するステップ202で始まる。例となる高帯域音声データ・セットは、参照文献[14]から[19]で記述されている。このステップは、図1のデータ入力ブロック180に対応する。

【0062】図6のステップ204で、音声データ・セットは、電話帯域幅を持つ間引かれた音声データ・セットを得るために間引かれる。これは、図1の機能ブロック182に対応する。間引かれた音声データ・セットは、入力音声データ・セットの高い方の帯域幅より低い帯域幅を持つであろう。間引き(decimation)プロセスは、参照[2]で記述されている。

【0063】次に、図6のステップ206で、帯域通過デジタル濾波器を間引かれた音声データ・セットに適用し、電話機器の伝送特性に特徴づける。これは、図1の機能ブロック182に対応する。これは、濾波された音声データ・セットを得るために行われる。帯域通過デジタル濾波器は、最大平坦設計アルゴリズムを持たなければならない。

【0064】次に、図6の中のステップ208で、その最大ダイナミック・レンジが非圧伸電話音声の最大レンジと一致するように、濾波された音声データ・セットの振幅が、再補正される。これは、図1の機能ブロック184に対応する。これは、振幅再補正音声データ・セットを得るために行われる。このステップの結果、その最大ダイナミック・レンジは非圧伸 μ -law電話音声の最大ダイナミック・レンジと一致し得る。代わりに、その最大ダイナミック・レンジは非圧伸A-law電話音声の最大ダイナミック・レンジと一致することもできる。

【0065】次に、図6ステップ210で、上記補正音声データ・セットを、電話中の音声信号の圧伸非圧伸の順序を表わしている量子化ノイズをもって修正する。これは、図1の機能ブロック186に対応する。これは、修正された音声データ・セットを得るために行われる。修正ステップは、 μ -lawノイズとしての量子化ノイズを持つことができる。代わりに、修正ステップは、A-lawノイズとしての量子化ノイズを持つことができる。

【0066】次に、図6のステップ212では、統計的パターン・マッチング・データ装置を訓練するために、音声認識プロセッサ50へ修正された音声データ・セットを入力する。これは、図1の出力データ・ブロック188に対応する。シミュレートされた電話チャネル音声185が、電話音声特有性を持つ電話コード・ブック105の特性を持つ音素モデル192を生成するために、音響的訓練プロセス190によって使われる。

【0067】次に、図6のステップ214で、たとえば、図5のPBX170からの信号のような、電話からの音声信号に対し、音声認識プロセッサ50を使って、音声認識が実行される。

【0068】電話チャネル・シミュレータ(ブロック185)を使用する高帯域音声の変換は、連続の音声認識装置に限られてなく、たとえば、IBM Tangora Dictation SystemおよびDragon Systems、ニュートン・マサチューセッツ、Dragon 30k DictateおよびKurzweil Applied Intelligence、Voice Report、Waltham、マサチューセッツおよび[20]で記述されるその他のシステム等のような様々な音声認識プロセッサに適用されるということに留意する必要がある。

10 【0069】上記本発明の説明において引用した参照文献は、以下の通りである。

【0070】[1] "Large Vocabulary Speaker and Dependent Continuous Speech Recognition: The Sphinx System"; Kai-Fu Lee; Carnegie Mellon University, Department of Electrical and Computer Engineering; April 1988; CMU-CS-88-148

[2] "A General Program to Perform Sampling Rate Conversion of Data by Rational Ratios"; from "Programs for Digital Signal Processing", Ed.: Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Processing Society; IEEE Press, 1979; Section 8.2, pp.8.2-1 to 8.2-7 by R.E. Crochiere

[3] "Theory and Application of Digital Signal Processing" L.R. Rabiner, B. Gold; Prentice Hall, 1975, pp. 91

[4] "Digital Processing of Speech Signals"; Prentice Hall Signal Processing Series; 1978, pp. 401-402, 411-413

30 [5] "An Algorithm for Vector Quantizer Design"; Y. Linde, A. Buzo, R. Gray, IEEE Transactions on Communications, Vol. com-28, no. 1, January 1980

[6] "IBM Continuous Speech Recognition System Programmers Guide"; B. Booth; 1992; currently unpublished, available on request.

[7] "Digital Telephony and Network Integration"; B. Keiser, E. Strange; Van Nostrand Reinhold Company Inc. 1985.; pp. 26-31

[8] "Design Subroutine (MAXFLAT) for Symmetric FIR Low Pass Digital Filters with Maximally-Flat Pass and Stop Bands" from "Programs for Digital Signal Processing", Ed.: Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Processing Society; IEEE Press, 1979; Section 5.3, pp. 5.3-1 to 5.3-6 by J. Kaiser

[9] "Acoustical and Environmental Robustness in Automatic Speech Recognition" A. Acero; Carnegie Mellon University, Department of Electrical and Computer Engineering; April 1990; CMU-CS-88-148

50 [10] "AIX Distributed Environments: NFS, NCS, RPC,

DS Migration, LAN Maintenance and Everything"; IBM International Technical Support Centers, Publication GG24-3489, May 8, 1990

[11] "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition"; L. Rabiner; Readings in Speech Recognition; Ed.: A. Waibel, K. Lee; Morgan Kaufmann; 1990; pp 267-296

[12] "IBM CallPath DirectTalk/2 General Information and Planning Manual"; International Business Machines publication no. GB35-4403-0; 1991

[13] "A Maximum Likelihood Approach to Continuous Speech Recognition"; L. R. Bahl, F. Jelinek, R. Mercer; Readings in Speech Recognition; Ed.: A. Waibel, K. Lee; Morgan Kaufmann; 1990; pp 308-319

[14] "Speech Corpora Produced on CD-ROM Media by The National Institute of Standards and Technology (NIST)", April, 1991

[15] "DARPA Resource Management Continuous Speech Database (RMI) Speaker Dependent Training Data", September 1989 NIST Speech Discs 2-1.1, 2-2.1 (2 Discs) NTIS Order No. PB89-226666

[16] "DARPA Resource Management Continuous Speech Database (RMI) Speaker-Independent Training Data", November 1989 NIST Speech Disc 2-3.1 (1 Disc) NTIS Order No. PB90-500539

[17] "DARPA Extended Resource Management Continuous Speech Speaker-Dependent Corpus (RM2)", September 1990 NIST Speech Discs 3-1.2, 3-2.2 NTIS Order No. PB90-501776

[18] "DARPA Acoustic-Phonetic Continuous Speech Corpus (TIMIT)", October 1990 NIST Speech Disc 1-1.1 NTIS Order No. PB91-0505065

[19] "Studio Quality Speaker-Independent Connected-Digit Corpus (TIDIGITS)", NIST Speech Discs 4-1.1, 4-2.1, 4-3.1 NTIS Order No. PB91-505592

[20] "The Spoken Word", Kai-Fu Lee, et al., Byte Magazine, July 1990, Vol- 15, No. 7; pp. 225-232

【0071】

【発明の効果】電話回線から入力される不特定の話し手の音声音声認識するシステムを構築することによって、たとえば、電話による顧客問い合わせ自動応答システムやレストラン電話案内など、従来技法では実現できなかった新たなコンピュータ適用業務を開発することができる。

【図面の簡単な説明】

【図1】電話チャネル・シミュレータ発明を含む、連続音声認識システムの論理的構造を図示する。

10 【図2】電話の符号器復号器インパルス応答を特徴づけるグラフである。

【図3】振幅特性韻文規格化ラジアン周波数を図示するグラフである。

【図4】対数振幅特性韻文規格化ラジアン周波数を図示するグラフである。

【図5】電話顧客業務通話センタにおける音声認識サーバのネットワークのブロック図である。

【図6】電話から得られる音声に応答するために音声認識装置を訓練するためのプロセスのステップ流れ図である。

【符号の説明】

100 アナログ・デジタル変換

104 ベクトル量子化

105 ベクトル量子化コードブック

192 音素モデル

135 語対文法

132 米語辞書

138 補助辞書

186 Muelawノイズ

30 186 Aelawノイズ

188 電話チャネル・シミュレータ

182 符号器デジタル復号・速度変換

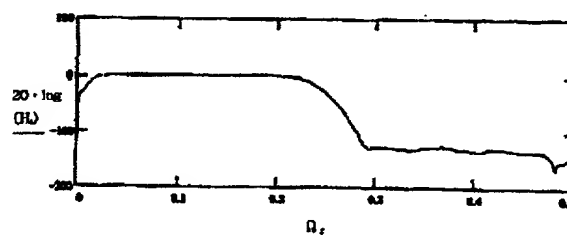
184 振幅補正(スケーリング)

134 タスク構築プログラム

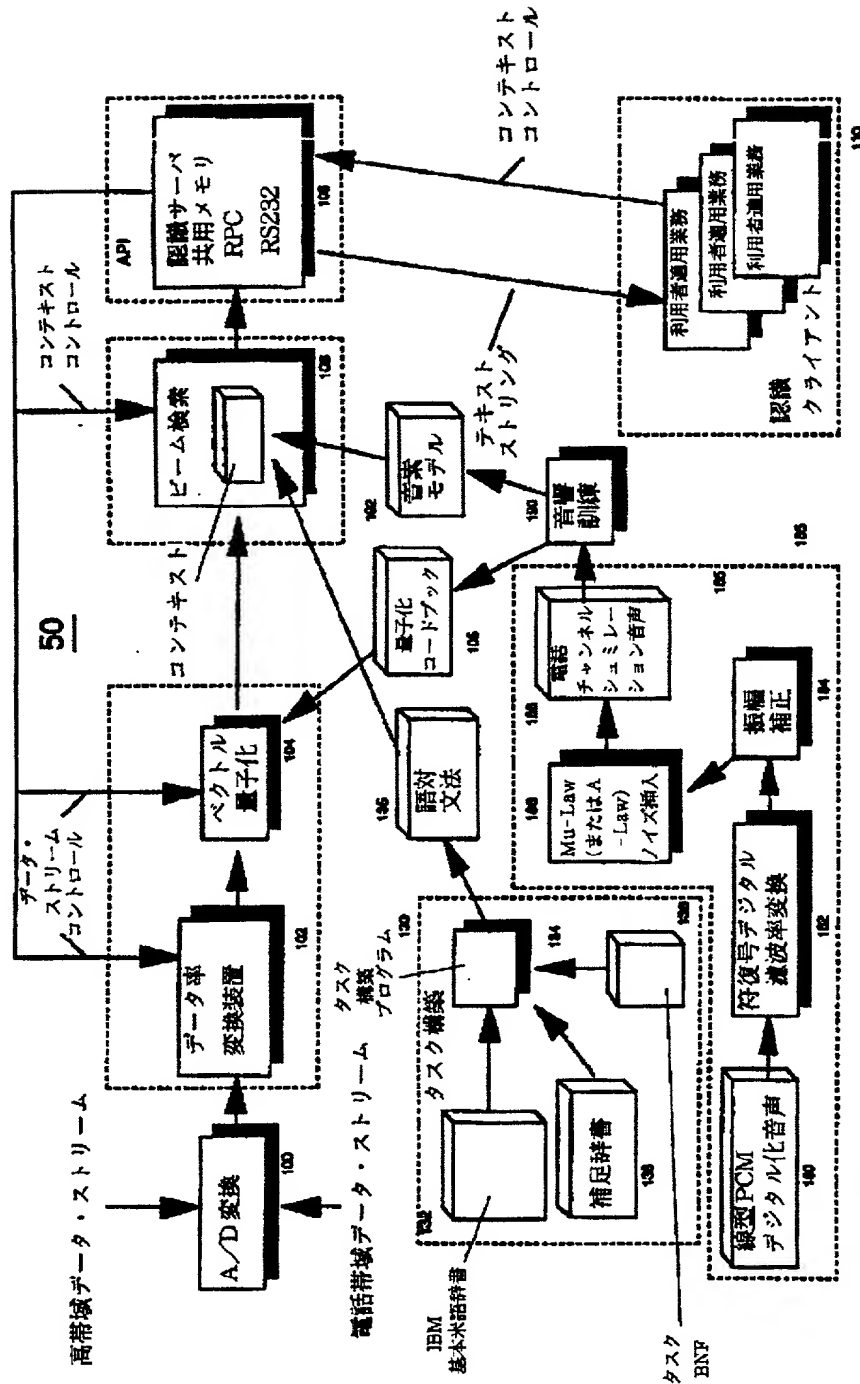
106 ビーム・サーチ

108 API (適用業務プログラム・インターフェース)

【図4】



【図1】



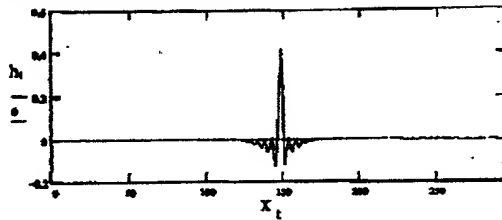
ICSRS システム論理構造

【図2】

```

N := READ (FilterSizeFile)  N = 295
t := 0..N-1
x_t := t
h_t := READ (FilterWeightFile)

```



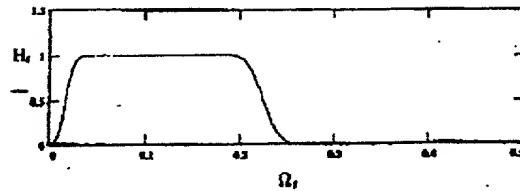
【図3】

$$F := 256$$

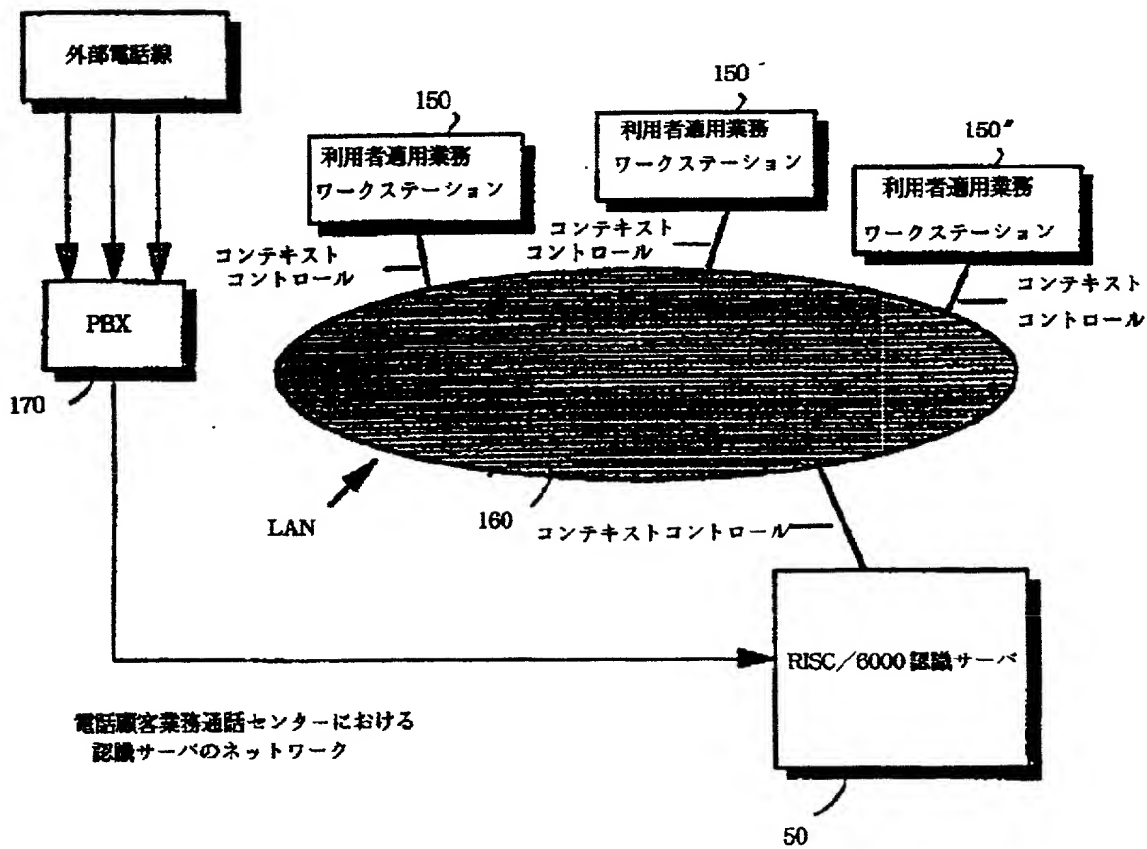
$$f := 0..F$$

$$\Omega_t := \frac{f}{2F}$$

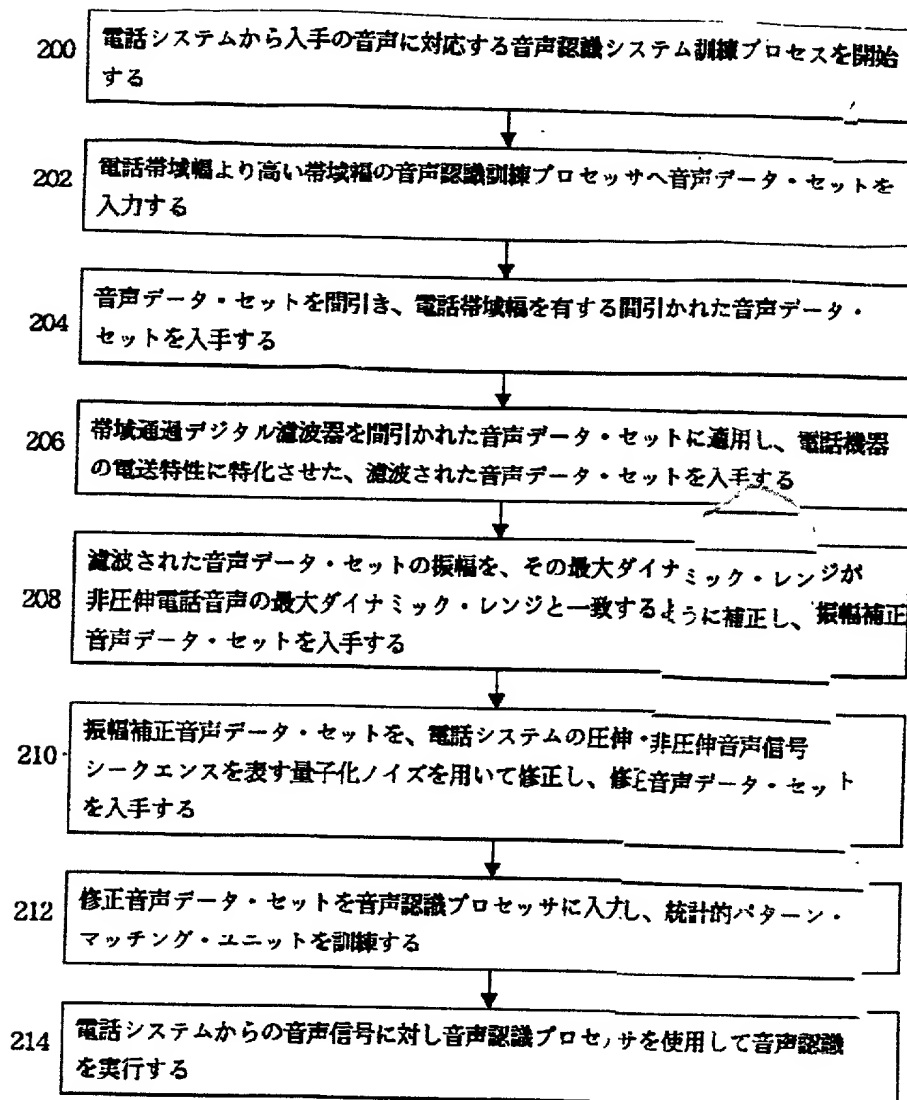
$$H_t := \left| \sum_{t} h_t \cdot e^{i \cdot 2 \cdot \pi \cdot t \cdot \frac{f}{2F}} \right|$$



【図5】



【図6】



フロントページの続き

(72)発明者 ノーマン エフ ブリックマン
 アメリカ合衆国メリーランド州ボトマック
 ミルバン ドライブ 11709番地



Consommation et
Affaires commerciales Canada

Consumer and
Corporate Affairs Canada

Bureau des brevets

Patent Office

Ottawa, Canada
K1A 0C9

(21) (A1) 2,091,658
(22) 1993/03/15
(43) 1994/09/16

(51) INTL. CL. ⁵ H04M-003/60; H04M-003/42

(19) (CA) APPLICATION FOR CANADIAN PATENT (12)

(54) Method and Apparatus for Automation of Directory
Assistance Using Speech Recognition

(72) Lennig, Matthew - Canada ;
Sharp, Robert D. - Canada ;
Bielby, Gregory J. - Canada ;

(73) Same as inventor

(57) 4 Claims

11017 U.S. PRO
09/930395
26,9/92
10/91/80

Notice: This application is as filed and may therefore contain an
incomplete specification.

Canada

CCA 3254 (10-92) 41 7530-21-036-3254

Abstract

In a telecommunication system, automatic directory assistance uses a voice processing unit comprising a database of vocabulary items and data representing a predetermined relationship between each vocabulary item and a calling number in a location served by the automated directory assistance apparatus. The voice processing unit issues messages to a caller making a directory assistance call to prompt the caller to utter a required one of said vocabulary items. The unit detects a calling number originating a directory assistance call and, responsive to the calling number and the relationship data computes a probability index representing the likelihood of a vocabulary item being the subject of the directory assistance call. The unit employs a speech recognizer to recognize, on the basis of the acoustics of the caller's utterance and the probability index, a vocabulary item corresponding to that uttered by the caller.

METHOD AND APPARATUS FOR AUTOMATION OF DIRECTORY ASSISTANCE USING
SPEECH RECOGNITION.

The invention relates to a method and apparatus for
providing directory assistance, at least partially automatically,
5 to telephone subscribers.

In known telephone systems, a telephone subscriber requiring
directory assistance will dial a predetermined telephone number.
In North America, the number will typically be 411 or 555 1212.
When a customer makes such a directory assistance call, the
10 switch routes the call to the first available Directory
Assistance (DA) operator. When the call arrives at the
operator's position, an initial search screen at the operator's
terminal will be updated with information supplied by the switch,
Directory Assistance Software (DAS), and the Operator Position
15 Controller (TPC). The switch supplies the calling number and the
DEMS call identifier, the DAS supplies the default locality and
zone, and the TPC supplies the default language indicator. While
the initial search screen is being updated, the switch will
connect the subscriber to the operator.

20 When the operator hears the "customer-connected" tone, the
operator will proceed to complete the call. The operator will
prompt for locality and listing name before searching the
database. When a unique listing name is found, the operator will
release the customer to the Audio Response Unit (ARU), which will
25 play the number to the subscriber.

Telephone companies handle billions of directory assistance
calls per year, so it is desirable to reduce labour costs by

minimizing the time for which a directory assistance operator is involved. As described in U.S. patent No. 5,014,303 (Velius) issued May 7, 1991, the entire disclosure of which is incorporated herein by reference, a reduction can be achieved by directing each directory assistance call initially to one of a plurality of speech processing systems which would elicit the initial directory assistance request from the subscriber. The speech processing system would compress the subscriber's spoken request and store it until an operator position became available, whereupon the speech processing system would replay the request to the operator. The compression would allow the request to be replayed to the operator in less time than the subscriber took to utter it.

Velius mentions that automatic speech recognition also could be employed to reduce the operator work time. In a paper entitled "Multiple-Level Evaluation of Speech Recognition Systems....", the entire disclosure of which is incorporated herein by reference, John F. Pitrelli et al discloses a partially automated directory assistance system in which speech recognition is used to extract a target word, for example a city name, from a longer utterance. The system strips off everything around the target word so that only the target word is played back to the operator. The operator initiates further action.

US patent No. 4,797,910 (Daudelin) issued January 10, 1989, the entire disclosure of which is incorporated herein by reference, discloses a method and apparatus in which operator involvement is reduced by means of a speech recognition system

which recognizes spoken commands to determine the class of call and hence the operator to which it should be directed. The savings to be achieved by use of Daudelin's speech recognition system are relatively limited, however, since it is not capable of recognizing anything more than a few commands, such as "collect", "calling card", operator", and so on.

These known systems can reduce the time spent by a directory assistance operator in dealing with directory assistance call, but only to a very limited extent.

10 An object of the present invention is to provide an improved automated directory assistance system capable of reducing, or even eliminating, operator involvement in directory assistance calls. To this end, in preferred embodiments of the present invention a speech recognition system elicits a series of
15 utterances by a subscriber and, in dependence upon a listing name being recognized, initiates automatic accessing of a database to determine a corresponding telephone number.

The system may be arranged to transfer or "deflect" a directory assistance call to another region when it recognizes
20 that the subscriber has uttered the name of a location which is outside its directory area.

Preferably, the system accesses the database taking account of a priori call distribution. A priori call distribution weights the speech recognition decision to take account of a
25 predetermined likelihood that a particular destination will be sought by a caller, conveniently based upon the caller's number.

According to one aspect of the invention, automated

directory assistance apparatus for at least partially automating
directory assistance in a telephone system comprises a voice
processing unit comprising a database of vocabulary items and
data representing a predetermined relationship between each
5 vocabulary item and a calling number in a location served by the
automated directory assistance apparatus, means for issuing
messages to a caller making a directory assistance call to prompt
the caller to utter a required one of said vocabulary items,
means for detecting a calling number originating a directory
10 assistance call, means responsive to a caller identifier, for
example the calling number, and said data for computing a
probability index representing the likelihood of a vocabulary
item being the subject of the directory assistance call, and
speech recognition means for recognizing, on the basis of the
15 acoustics of the caller's utterance and the probability index,
a vocabulary item corresponding to that uttered by the caller.

Embodiments of the invention may comprises means for
prompting a subscriber to specify a location, means for detecting
a place name uttered in response, means for comparing the uttered
20 place name with a database and independence upon the results of
the comparison selecting a message, playing the message to the
subscriber. If the place name has been identified precisely as
a city or location name, the message may be an NPA.

Alternatively the message could be to the effect that the
25 number is in a different calling or directory area and offer to
give the subscriber the area code. In that case, the speech
recognition system would be capable of detecting a positive

answer and supplying the appropriate area code from the data base. Another variation is that the customer could be asked if the call should be transferred to the directory assistance in the appropriate area. If the subscriber answered in the affirmative,
5 the system would initiate the call transfer.

As mentioned the recognition system preferably makes its choice based upon a predetermined probability index derived using an identifier such as calling number. The probability index will bias the selection in favour of, for example, addresses in the
10 same geographical area, such as the same city.

The probability index need not be geographical, but might be temporal, perhaps according to time-of-day, or week or year. For example, certain businesses, such as banks, are unlikely to be called at one o'clock in the morning whereas taxi firms are.
15 Likewise, people might call a ski resort in winter but not in summer. Hence the nature of the business can be used to weight the selection of certain portions or segments of the data base for a particular enquiry.

The discourse between the speech recognition system and the
20 subscriber may be recorded. If the system disposes of the call entirely without the assistance of the operator, the recording could be erased immediately. On the other hand, if the call cannot be handed entirely automatically, at the point at which the call is handed over to the operator, the recording of the
25 entire discourse, or at least the subscriber's utterances, could be played back to the operator. Of course, the recording could be compressed using the prior art techniques mentioned above.

According to a second aspect of the invention, a method of at least partially automating directory assistance in a telephone system comprises a voice processing unit comprising a database of vocabulary items and data representing a predetermined relationship between each vocabulary item and a calling number in a location served by the automated directory assistance apparatus, comprises the steps of issuing messages to a caller making a directory assistance call to prompt the caller to utter a required one of said vocabulary items, detecting a calling number originating a directory assistance call, computing, in response to the calling number and said data, a probability index representing the likelihood of a vocabulary item being the subject of the directory assistance call, and employing speech recognition means to recognize, on the basis of the acoustics of the caller's utterance and the probability index, a vocabulary item corresponding to that uttered by the caller.

An embodiment of the invention will now be described by way of example only and with reference to the accompanying drawings in which:

Figure 1 is a general block diagram of a known telecommunications system;

Figure 2 is a simplified block diagram of parts of a telecommunications system employing an embodiment of the present invention;

Figures 3A and 3B are a general flow chart illustrating the processing of a directory assistance call in the system of Figure 2;

Figure 4 is a chart illustrates the frequency with which certain cities are requested by callers the same or other cities; and

Figure 5 is a graph of call distribution according to distance and normalized for population of the called city.

Figure 1 is a block diagram of a telecommunications system as described in US patent number 4,797,910. As described therein, block 1 is a telecommunications switch operating under stored program control. Control 10 is a distributed control system operating under the control of a group of data and call processing programs to control various parts of the switch. Block 12 is a voice and data switching network capable of switching voice and/or data between inputs connected to the switching network. An automatic voice processing unit 14 is connected to the switching network 12 and controlled by control 10. The automated voice processing unit receives input signals which may be either voice or dual tone multifrequency (DTMF) signals and is capable of determining whether or not the DTMF signals are allowable DTMF signals and initiating action appropriately. In the system described in US patent number 4,797,910, the voice processing unit has the capability to distinguish among the various elements of a predetermined list of spoken responses. The voice processing unit 14 also has the capability to generate tones and voice messages to prompt a customer to speak or key information into the system for subsequent recognition by the voice recognition unit. In addition, the voice processing unit 14 is capable of recording

a short customer response for subsequent playback to a called terminal. The voice processing unit 14 generates an output data signal, representing the result of the voice processing. This output data signal is sent to control 10 and used as an input to

5 the program for controlling establishment of connections in switching network 12 and for generating displays for operator position 24. In order to set up operator assistance calls, switch 1 uses two types of database system. Local data base 16 is directly accessible by control 10 via switching network 12.

10 Remote data base system 20 is accessible to control 10 via switching network 12 and interconnecting data network 18. A remote data base system is typically used for storing data that is shared by many switches. For example, a remote data base system might store data pertaining to customers for a region; the

15 particular remote data base system that is accessed via data network 18 would be selected to be the remote data base associated with the region of the called terminal. Interconnecting data network 18 can be any well known data network and specifically could be a common channel signalling

20 system such as the international standard telecommunications signalling system CCS 7.

Transaction recorder 22 is used for recording data about calls for subsequent processing. Typically, such data is billing data. The transaction recorder 22 is also used for recording

25 traffic data in order to engineer additions properly and in order to control traffic dynamically.

5 The present invention will be employed in a telecommunications system which is generally similar to that described in US patent number 4,797,910. Figure 2 is a simplified block diagram of parts of the system involved in a directory assistance call, corresponding parts having the same reference numbers in both Figure 1 and Figure 2. As shown in Figure 2, block 1 represents a telecommunications switch operating under stored program control provided by a distributed control system operating under the control of a group of data and call processing programs to control various parts of the switch. The switch 1 comprises a voice and data switching network 12 capable of switching voice and/or data between inputs and outputs of the switching network. As an example, Figure 1 shows a trunk circuit 31 connected to an input of the network 12. A caller's station apparatus or terminal 40 is connected to the trunk circuit 31 by way of network routing/switching circuitry 30 and an end office 33. The directory number of the calling terminal, identified, for example, by automatic number identification, is transmitted from the end office switch 33 connecting the calling terminal 40 to switch 1.

10 An operator position controller 23 connects a plurality of operator positions 24 to the switch network 12. A data/voice link 27 connects an automated voice processing unit 14A to the switching network 12. The automated voice processing unit 14A will be similar to that described in US patent number 4,797,910 in that it is capable of generating tones and voice messages to prompt a customer to speak or key dual tone multifrequency (DTMF)

information into the system, determine whether or not the DTMF signals are allowable DTMF signals, and initiating action appropriately and to apply speech recognition to spoken inputs.

In addition, the voice recognition unit 14A is capable of

5 recording a short customer response for subsequent playback to a human operator. Whereas in US patent number 4,797,910, however, the voice processing unit 14 merely has the capability to distinguish among various elements of a very limited list of spoken responses to determine the class of the call and to which
10 operator it should be directed, voice processing unit 14A of Figure 2 is augmented with software enabling it to handle a major part, and in some cases all, of a directory assistance call.

Each operator position 24 comprises a terminal which is used by an operator to control operator assistance calls. Data
15 displays for the terminal are generated by operator position controller 23.

In order to provide the enhanced capabilities needed to automate directory assistance calls, at least partially, the voice processing unit 14A will employ flexible voice recognition
20 technology and *a priori* probabilities. For details of a suitable flexible voice recognition system the reader is directed to Canadian patent application number 2,069,675 filed May 27, 1992, the entire disclosure of which is incorporated herein by reference. A priori probability uses the calling number to
25 determine a probability index which will be used to weight the speech recognition based upon the phonetics of the caller's utterances. The manner in which the *a priori* probabilities are

determined will be described in more detail later with reference to Figures 4 and 5.

As shown in Figure 2, in embodiments of the present invention, when the voice processing unit 14 receives a directory assistance call, it determines in step 301 whether or not the number of the calling party is included. If it is not, the voice processing unit immediately redirects the call for handling by a human operator in step 302. If the calling number is included, in step 303 the voice processing unit issues a bilingual greeting message to prompt the caller for the preferred language. At the same time, the message may let the caller know that the service is automated, which may help to set the caller in the right frame of mind. Identification of language choice at the outset determines the language to be used throughout the subsequent process, eliminating the need for bilingual prompts throughout the discourse and allowing the use of less complexity in the speech recognition system.

If no language is selected, or the answer is unrecognizable, the voice processing unit 14 hands off the call to a human operator in step 304 and plays back to the operator whatever response the caller made in answer to the prompt for language selection. It will be appreciated that the voice processing unit 14 records at least the caller's utterances for subsequent playback to the operator, as required.

If the caller selects French or English, in step 305 the voice processing unit 14 uses the calling number to set a priori

probabilities to determine the likelihood of certain locality names being requested. The voice processing unit has a basic vocabulary of localities, e.g. numbering plan areas (NPA) which it can recognize and a listing of latitudes and longitudes for determining geographical location for calling numbers. In step 305, the voice processing unit computes probabilities for the entire vocabulary based upon distance from the locality of the calling number and population and also within the calling number's own area code or locality. In step 306, the voice processing unit issues the message "For what city?" to prompt the caller to state the name of the city, identifying the locality, and tries to recognize the name from its vocabulary using speech recognition based upon the acoustics, as described in the aforementioned Canadian patent application number 2,069,675. The voice processing unit will use the a priori probabilities to influence or weight the recognition process. If the locality name cannot be recognized, decision steps 307 and 308 cause a message to be played, in step 309, to prompt the caller for clarification. The actual message will depend upon the reason for the lack of recognition. For example, the caller might be asked to simply speak more clearly. Decision step 308 permits a limited number of such attempts at clarification before handing the call off to a human operator in step 310. The number of attempts will be determined so as to avoid exhausting the caller's patience.

If the locality name is recognized, the voice recognition unit determines in step 311 whether or not the locality is served

by the directory assistance office handling the call. If it is not, the voice processing unit will play a "deflection" message instep 312 inviting the caller to call directory assistance for that area. It is envisaged that, in some embodiments of the invention, the deflection message might also give the area code for that locality and even ask the caller if the call should be transferred.

If the requested locality is served by the directory assistance office handling the call, in step 313 the voice processing unit will transmit a message asking the caller to state whether or not the called party has a business listing and employs speech recognition to recognize the caller's response. If the response cannot be recognized, decision steps 314 and 315 and step 316 will cause a message to be played to seek clarification. If a predetermined number of attempts at clarification have failed to elicit a recognizable response, decision step 315 and step 317 hand the call of to a human operator. If a response is recognized in step 314, decision step 318 determines whether or not a business was selected. If not, step 319 plays the message "for what listing?" and, once the caller's response has been recorded, hands off to the human operator.

If decision step 318 indicates that the required number is a business listing, in step 320 the voice processing unit plays a message "For what business name?" and employs speech recognition to recognize the business name spoken by the caller in reply. Once again, the recognition process involves an

acoustic determination based upon the phonetics of the response and a *priori* probabilities.

If the business name cannot be recognized, in steps 321, 322 and 323 the unit prompts the caller for clarification and, as
 5 before, hands off to a human operator in step 324 if a predetermined number of attempts at clarification fail.

It should be noted that, when the unit hands off to a human operator in step 310, 317, 319 or 324, the operator's screen will display whatever data the automatic system has managed to
 10 determine from the caller and the recording of the caller's responses will be replayed.

If the unit recognizes the business name spoken by the caller, in step 325 the unit determines whether or not the data base lists a main number for the business. If not, the unit
 15 hands off to the human operator in step 326 and language, locality and selected business will be displayed on the operator's screen. If there is a main number for the business, in step 327 the unit plays a message asking if the caller wants the main number and uses speech recognition to determine the
 20 answer. If the caller's response is negative, step 328 hands off to the human operator. If the caller asks for the main number, however, in step 329 the unit instructs the playing back of the main number to the caller, and terminates the interaction with the caller.

25 As mentioned earlier, the use of a *priori* probabilities enhances the speech recognition capabilities of the voice processing unit 14A. Statistics collected from directory

assistance data show a relation between the origin of a call and its destination. An *a priori* model of probability that a person at a phone number NPA/NXX asks for a locality l_j , is an additional piece of information which improves the recognition performance. The *a priori* model expresses the probability

$P(l_j|l_i)$ of someone calling from locality l_i and requesting a locality l_j . The probability $P(l_j|l_i)$ depends on the population

of l_i and the distance between l_j and l_i . The input call locality l_i is not known precisely. From the input phone number NPA/NXX, the Central Office (CO) may be identified using the Bellcore mapping. Following that step, a set of input localities related to that Central Office is considered. The probability of calling a locality l_j from a phone number NPA/NXX is:

$$P(l_j|NPA/NXX) = \sum_{l_i \in CO} P(l_i) P(l_j|l_i) \quad (\text{Eq 1})$$

The probability $P(l_i)$ of each calling locality l_i associated with a CO is proportional to its population. Finally, the total recognition score for each locality is a weighted sum of the usual acoustic likelihood $\log P(Y_1 Y_2 \dots Y_N | O_j)$ and the logarithm of

$P(O_j(l_j) | NPA/NXX)$:

$$\text{Score}(O_j(l_j)) = \log P(Y_1 Y_2 \dots Y_M | O_j) + \lambda \log P(O_j(l_j) | N_{panxxx}) \quad \text{EQ 2}$$

where O_j is the orthography of the location l_j . An a priori model may be arranged to distinguish between populations having French or English as their first language. Knowing the language selected by the user, the population using that language is used to estimate $P(l_j | l_i)$. A minimum value of 10% of the population

is used to avoid excessively penalizing a language.

As an example, an a priori probability model developed using directory assistance data collected in the 514, 418 and 819 area codes, is shown graphically in Figure 4. In each of these area codes, the number of requests to and from each NXX was collected; faint lines appear indicating the frequency of "any city requesting Montreal", "Montreal requesting any city", and "any city requesting itself". From these data it was possible to estimate the parameters of a parametric model predicting the probability of a request for information being made for any target city given the calling (NPA) NXX. The parameters of the model proposed are the called city's population and the distance between the two cities. Where o is the originating locality, d is the destination locality, and S is the size of a locality, then the likelihood of a request about d given o is

$$L(d|o) = S(d) * f(|\vec{o} - \vec{d}|)$$

The normalized likelihood is

$$\bar{L}(d|o) = 0.60 \frac{L(d|o)}{\sum_{\text{over all } d'} L(d'|o)}$$

When the destination city is also the origin city, the likelihood is higher, so this is treated as a special case.

It is assumed that 60% of DA requests are placed to localities including the originating locality as governed by the equation above, and an additional 40% of DA request are for the originating city, giving

$$P(d|o) = \begin{cases} \bar{L}(d|o), & d \neq o \\ \bar{L}(d|o) + 0.40, & d = o \end{cases}$$

Intuitively, the function $f(o,d)$ varies inversely with the distance between cities. In order to better define the function, a table of discrete values for certain distance ranges was derived from community of interest data collected in the three Quebec area codes. The distance units used in this section are the ones used by Bellcore to maintain geographical locality coordinates in North America. One kilometre is roughly equal to 1.83 Bellcore units.

The discrete function values f computed for a given distance range in the province of Quebec are given in the Table below for each area code. Since the goal was to obtain an *a priori* model for the entire province, the values for $f(o,d)$ were computed for the province as a whole through factoring in the probability of a call originating in each area code. This was estimated to be

in proportion to the number of NXX's per area code relative to the province as a whole.

This gives

$$f(\text{Province}) = \{0.40 f(514)\} + \{0.27 f(819)\} + \{0.33 f(418)\}$$

5

distance	514	819	418	Province
0-25	1.0	1.0	1.0	1.00
26-50	0.9	0.3	0.7	0.67
51-75	0.4	0.0	0.2	0.23
76-100	0.1	0.0	0.3	0.14
101-125	0.1	0.0	0.1	0.07
126-150	0.1	0.0	0.1	0.07
151-175	0.0	0.0	0.2	0.07
176-200	0.0	0.0	0.0	0.00
>200	0.0	0.0	0.0	0.00

10

15

Given the sparseness of data, the model for obtaining weights as a function of distance was converted from nonparametric to parametric. For this purpose, a least square fit was performed on the data from ranges 26-50 to 151-175. The distance value used in the fitting process was the median distance in each range. An analysis of various function forms for the regression showed that the form below provided the closest fit to the collected data:

20

$$f'(\text{distance}) = \{A/\text{distance}\} + B$$

The best coefficients obtained from the analysis were

$$A = 33.665$$

$$B = -0.305$$

5 This function reaches zero when the distance is equal to
196. In order to not eliminate the possibility of handling a DA
request when the distance was greater than this value, the
function was modified to accommodate distances of up to twice the
maximum distance between any pair of cities with population
10,000 or greater in the province. The two most distant cities
10 that matched this criteria were Rouyn-Noranda and Gaspé at a
distance of 2,103 units. The maximum distance at which f would
be zero was set to be 4,207 distance units. The function
switches to a negative slope linear function at the point where
15 the predicted value of f is 0.01. This corresponds to a distance
value of 167.

The final f becomes

$$\min(1, (33.65/d) - 0.305, d \leq 167$$

$$0.01, d > 167$$

20 The fit of this model to the collected data, labelled
"nonparametric model:", is shown in Figure 5.

In order to determine the effects of the *a priori* model on
recognition rate, the model was applied to simulated DA requests,
25 and each token in the test set was rescored to take a *priori*
likelihood into account. The function used for rescoring was

$$\text{weighted score} = nas + K \log\{P(o|d)\},$$

where nas is the normalized acoustic score, the acoustic score over the number of frames in the utterance. The proportionality constant K was trained to maximize the recognition rate over the province of Quebec. The distribution of tokens in the test set

5 is normalized to be that predicted by the *a priori* model. For this reason a correctly recognised simulated DA request from a city to the same city carries more weight when computing recognition rate than does a request for a small distant city with a correspondingly low *a priori* probability. A recognition

10 rate was thus determined per city and then the overall provincial recognition rate was computed by taking the sum of the rate for all cities in proportion to their respective populations. The only assumption made in applying the model was that the calling NPA/NXX is known, which allows the utterance to be rescored by

15 mapping it to all cities corresponding to the given entry in the Bellcore database.

The *a priori* model was further refined in order to avoid favouring the bigger cities unduly, as the recognition rate on these based on acoustics alone was already above average. For

20 this purpose, constants were introduced in the model corresponding to population ranges over the target cities in order to reduce the effective populations. These constants were not applied to the modelled distribution of requests since this would invalidate the method for computing the provincial

25 recognition rate. The function defining likelihood becomes

$$L(d|o) = K_{r(d)} S(d) f(|\vec{o} - \vec{d}|)$$

where $r(d)$ is a range of destination locality population for which the constant K applies. The best ranges and their associated constants were then determined empirically from a development set.

5 Thus, using a *a priori* call distribution, and flexible voice recognition, embodiments of the present invention are capable of automating at least the front end of a directory assistance call and in some cases the entire call.

10 The embodiment of the invention described above is by way of example only. Various modifications of, and alternatives to, its features are possible without departing from the scope of the present invention. For example, the voice processing system might be unilingual or multilingual rather than bilingual. The *a priori* probabilities need not be geographical but might be
15 determined in other ways. For example, they might be determined according to time-of-day or season of year, or determined with reference to a history of calls placed by a particular caller or callers.

THE EMBODIMENTS OF THE INVENTION IN WHICH AN EXCLUSIVE RIGHT OR PRIVILEGE IS CLAIMED ARE DEFINED AS FOLLOWS:

- 5 1. Apparatus for at least partially automating directory assistance in a telephone system, comprising a voice processing unit comprising a database of vocabulary items and data representing a predetermined relationship between each vocabulary item and a calling number in a location served by the automated
- 10 directory assistance apparatus, means for issuing messages to a caller making a directory assistance call to prompt the caller to utter a required one of said vocabulary items, means for detecting a calling number originating a directory assistance call, means responsive to the calling number and said data for
- 15 computing a probability index representing the likelihood of a vocabulary item being the subject of the directory assistance call, and speech recognition means for recognizing, on the basis of the acoustics of the caller's utterance and the probability index, a vocabulary item corresponding to that uttered by the
- 20 caller.
2. Apparatus as claimed in claim 1, further comprising means for transmitting a message to the caller giving the required directory number corresponding to the vocabulary item.
- 25 3. Apparatus for at least partially automating directory assistance in a telephone system, including voice processing

means for issuing to a directory assistance caller a message inviting the caller to utter the name of a location, recognizing the place name from the utterance, determining whether or not the location is within the area served by the automatic directory assistance apparatus and, in the event that it is not, playing a message to the caller inviting the caller to direct the directory assistance request to an alternative locality.

4. A method of at least partially automating directory assistance in a telephone system comprises a voice processing unit comprising a database of vocabulary items and data representing a predetermined relationship between each vocabulary item and a calling number in a location served by the automated directory assistance apparatus, comprises the steps of issuing messages to a caller making a directory assistance call to prompt the caller to utter a required one of said vocabulary items, detecting a calling number originating a directory assistance call, computing, in response to the calling number and said data, a probability index representing the likelihood of a vocabulary item being the subject of the directory assistance call, and employing speech recognition means to recognize, on the basis of the acoustics of the caller's utterance and the probability index, a vocabulary item corresponding to that uttered by the caller.

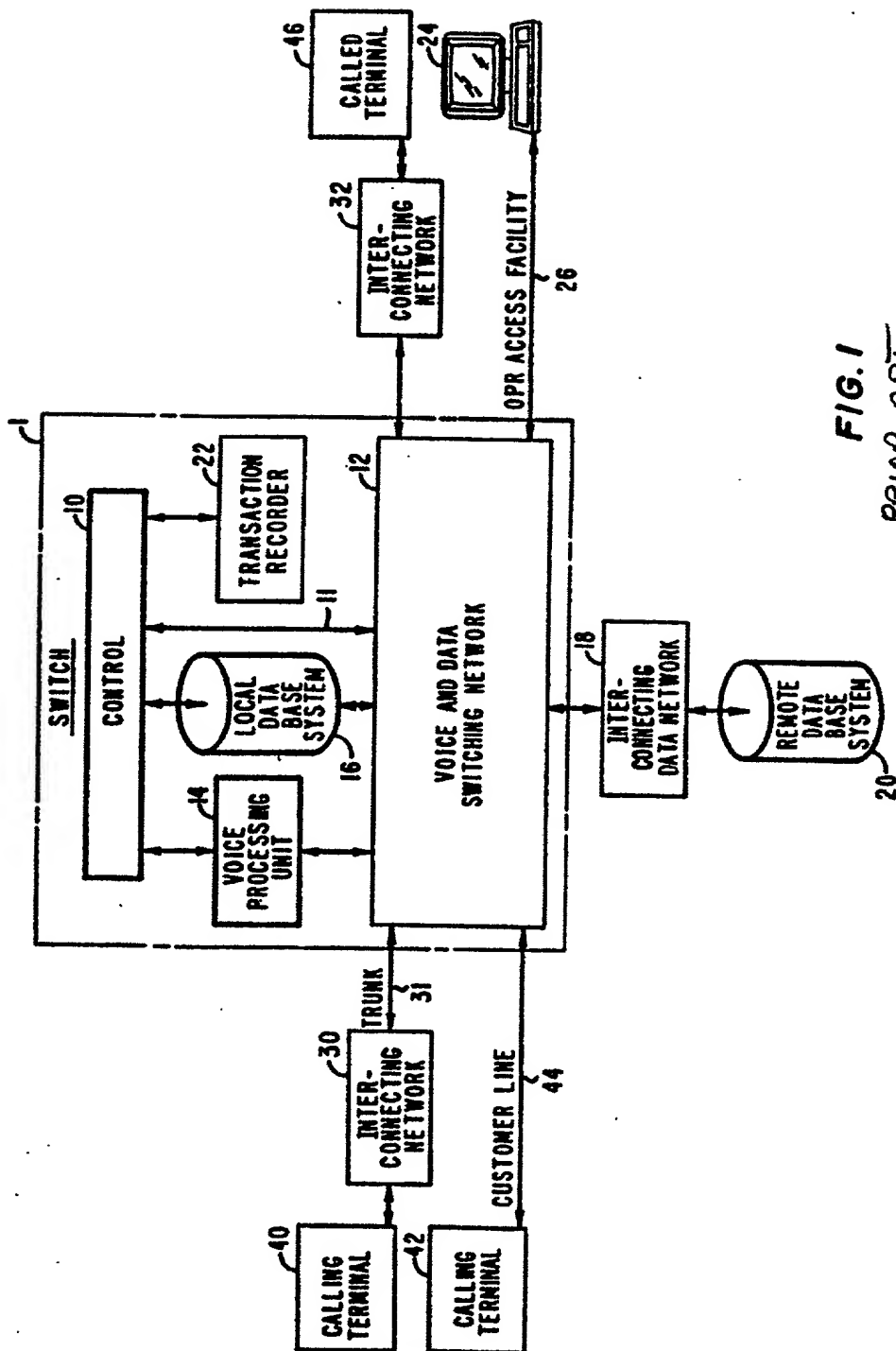


FIG. 1
PRIOR ART

Thomas Adams & Sons
AGENT FOR APPLICANT

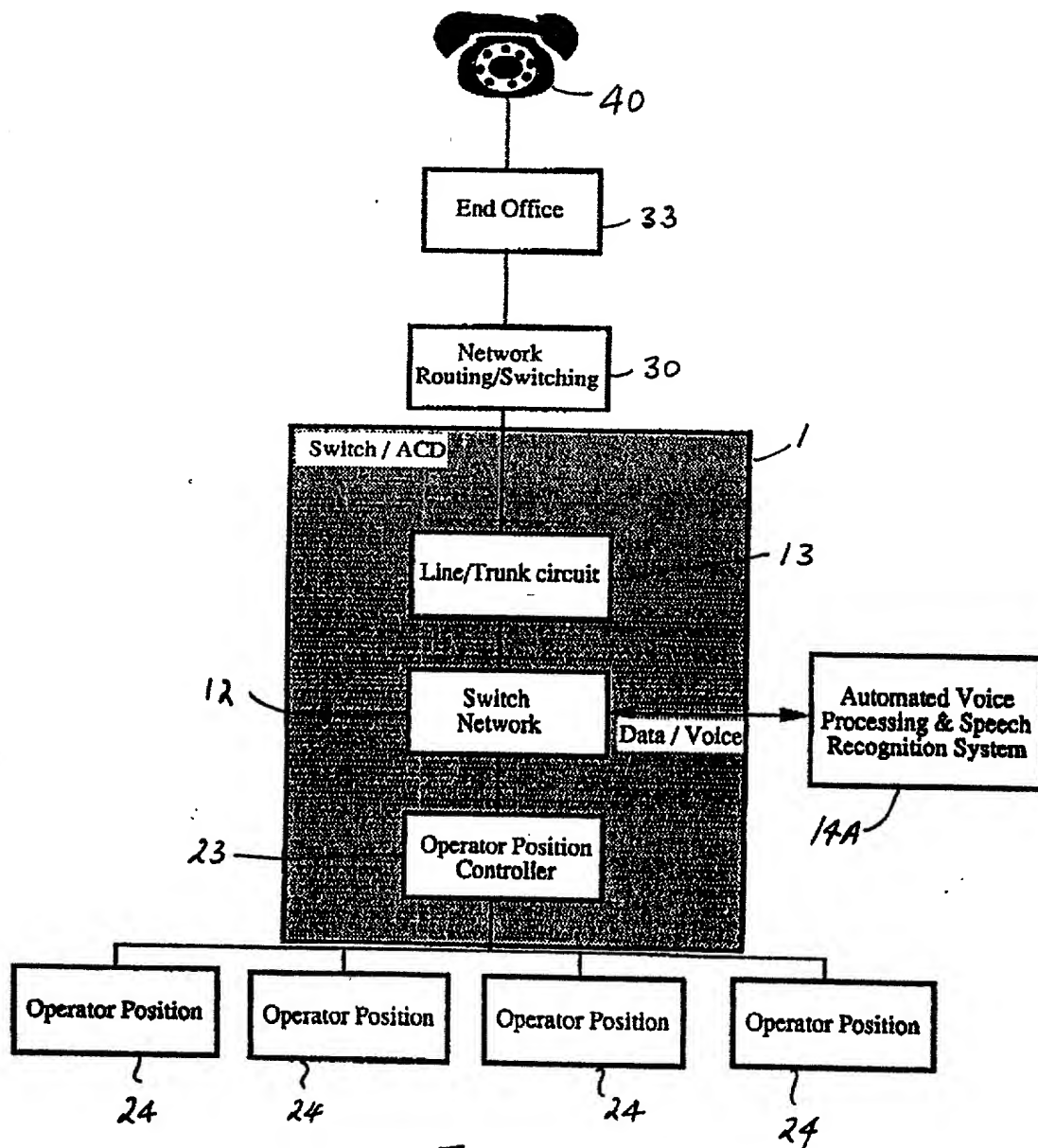


FIG. 2

Thomas Adam & Associates
AGENT FOR APPLICANT

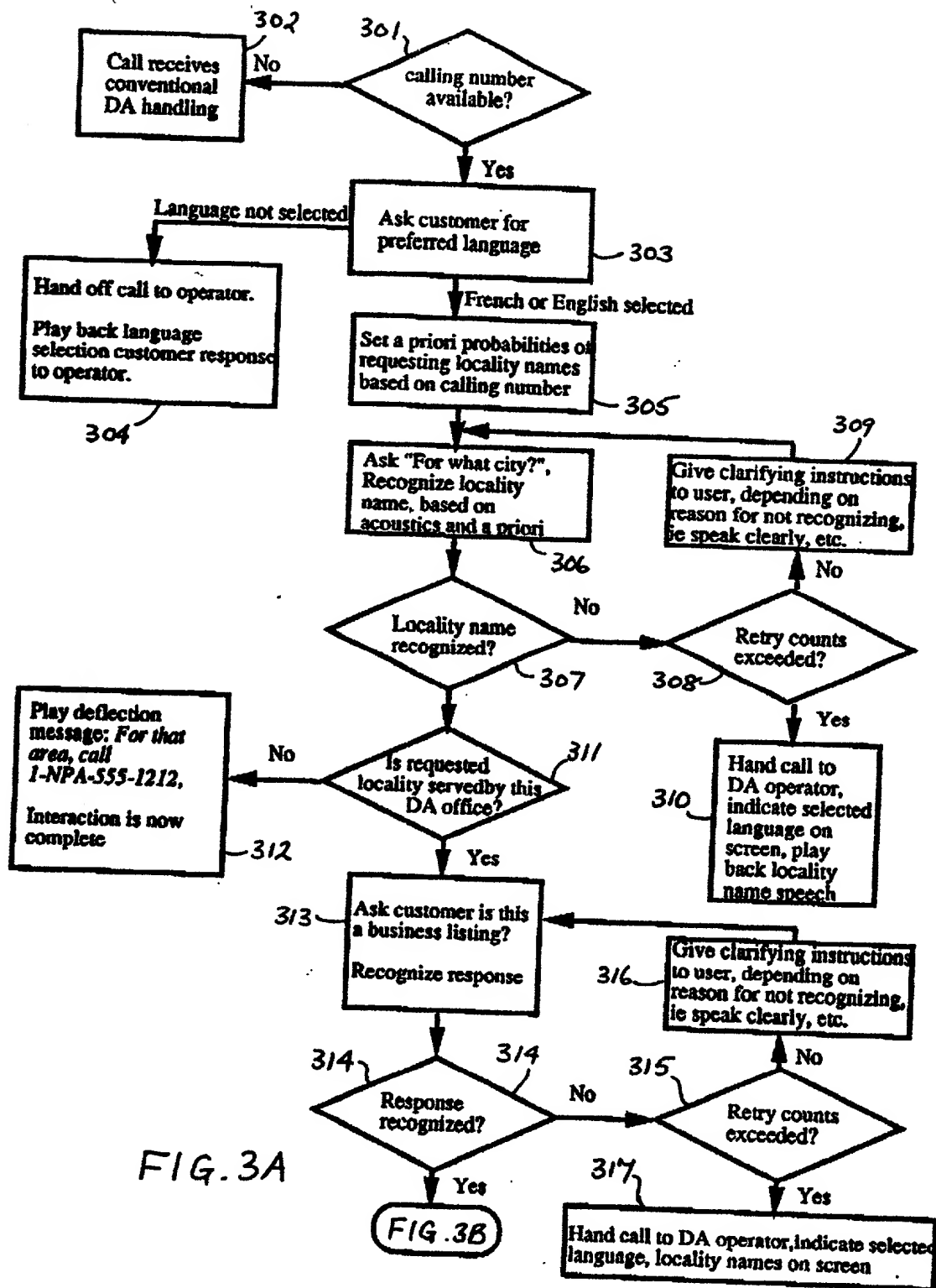
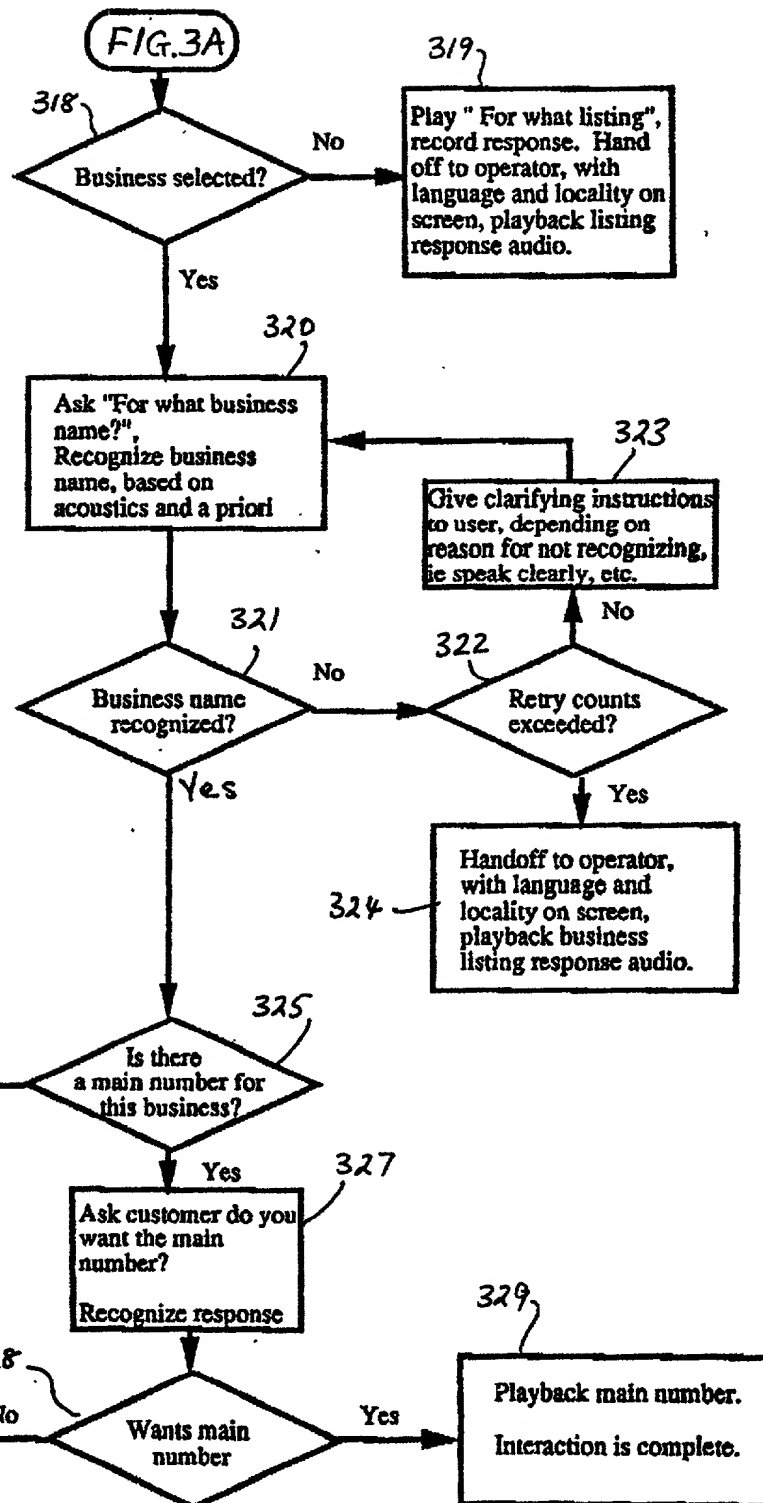


FIG. 3A

FIG. 3B

Thomas Adams & Assoc
AGENT FOR APPLICANT



Thomas Adams & Associates
AGENT FOR APPLICANT

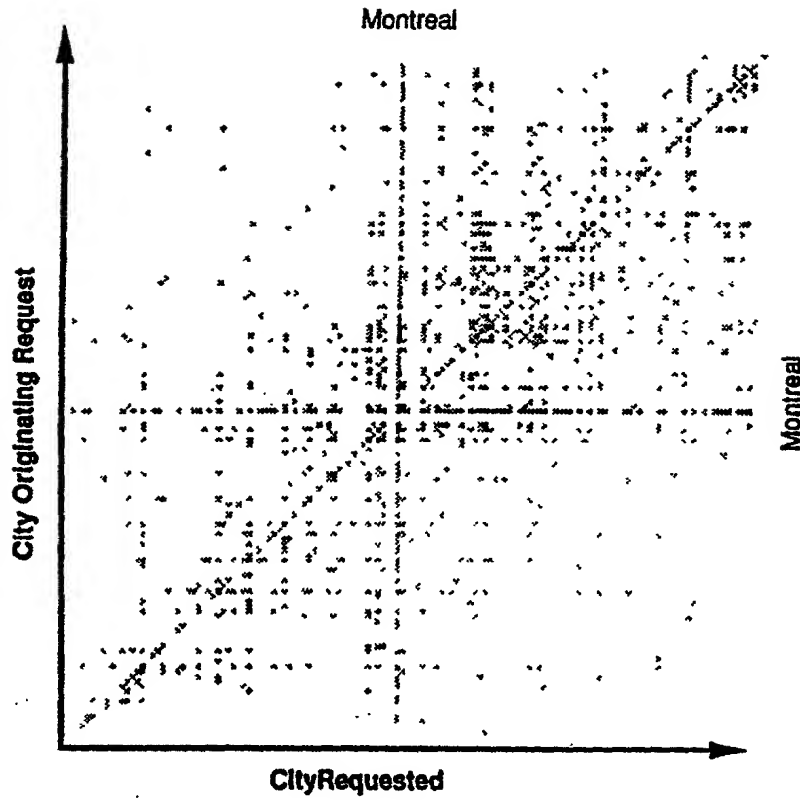
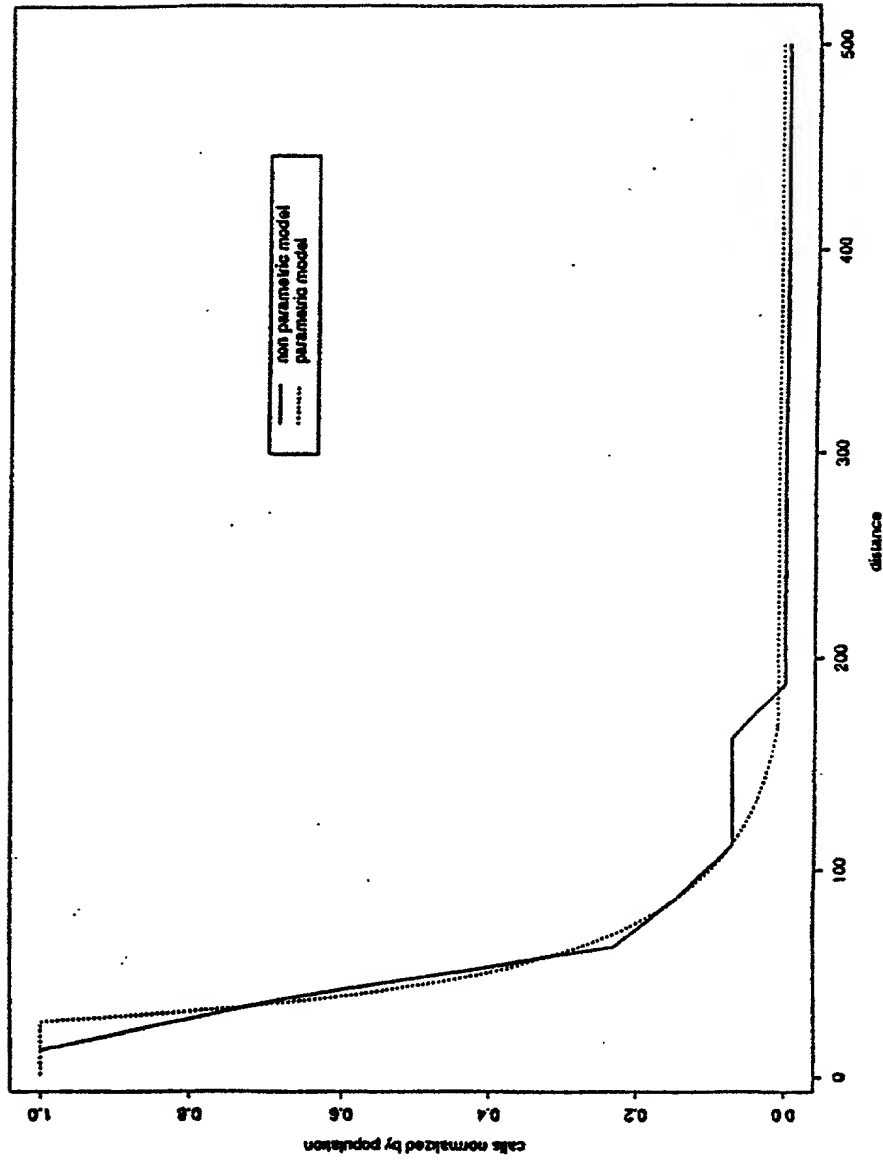


FIG. 4

Thomas Adams & Assoc.
AGENT FOR APPLICANT

FIG. 5



Thomas Adams & Assoc.
AGENT FOR APPLICANT

Requested Patent: WO9305605A1

Title: METHOD AND SYSTEM FOR HOME INCARCERATION ;

Abstracted Patent: US5170426 ;

Publication Date: 1992-12-08 ;

Inventor(s):

D ALESSIO FREDERICK D (US); KEOPPE ALFRED C (US); WEGLEITNER MARK A (US); MCALLISTER ALEXANDER I (US) ;

Applicant(s): BELL ATLANTIC NETWORK SERVICES (US) ;

Application Number: US19910758051 19910912 ;

Priority Number(s): US19910758051 19910912 ;

IPC Classification: G08B23/00 ; G08B26/00 ; H04M11/04 ;

Equivalents: AU2574792, NZ244333

ABSTRACT:

A method and system is disclosed for remotely verifying attendance of a particular person at a predetermined confined area. Monitoring and verification is performed through a telephone network including a telephone on the premises of the location of confinement and a control center. Voice verification, using voice analysis of speech transmitted in a telephone call from the site to the center is performed during periodic testing. A voice template vocabulary is established for the individual and used for voice verification. Caller line identification of each incoming call is performed to verify that call originates from the appropriate location. The confined individual is required, either randomly or at scheduled intervals, by the system to call the control center and recite a statement including randomly selected words from the template vocabulary.

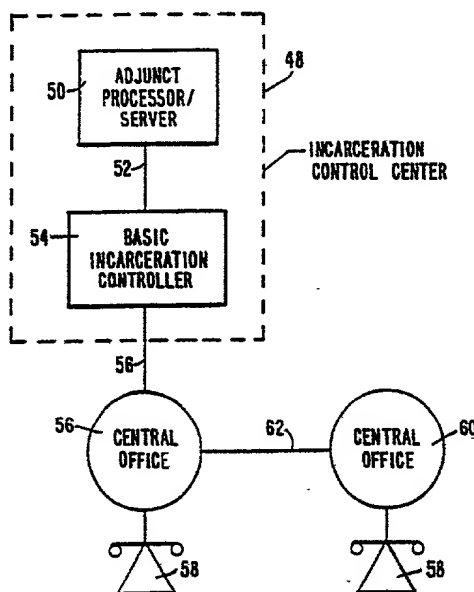
2025 RELEASE UNDER E.O. 14176



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 5 : H04M 11/04		A1	(11) International Publication Number: WO 93/05605
			(43) International Publication Date: 18 March 1993 (18.03.93)
(21) International Application Number: PCT/US92/07645 (22) International Filing Date: 11 September 1992 (11.09.92) (30) Priority data: 758,051 12 September 1991 (12.09.91) US (71) Applicant: BELL ATLANTIC NETWORK SERVICES, INC. [US/US]; 1310 North Court House Road, Arlington, VA 22201 (US). (72) Inventors: D'ALESSIO, Frederick, D. ; 808 Polo Place, Great Falls, VA 22066 (US). WEGLEITNER, Mark, A. ; 1119 North Rochester Street, Arlington, VA 22205 (US). MCALLISTER, Alexander, I. ; 12711 Gould Road, Wheaton, MD 20906 (US). KEOPPE, Alfred, C. ; No. 2 East Acres Drive, Pennington, NJ 08534 (US).		(74) Agent: LEBLANC, Robert, E.; Lowe, Price, Leblanc & Becker, 99 Canal Center Plaza, Suite 300, Alexandria, VA 22314 (US). (81) Designated States: AT, AU, BB, BG, BR, CA, CH, CS, DE, DK, ES, FI, GB, HU, JP, KP, KR, LK, LU, MG, MN, MW, NL, NO, PL, RO, RU, SD, SE, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, SN, TD, TG). Published <i>With international search report.</i>	

(54) Title: METHOD AND SYSTEM FOR HOME INCARCERATION



(57) Abstract

A method and system is disclosed for remotely verifying attendance of a particular person at a predetermined confined area. Monitoring and verification is performed through a telephone network including a telephone (58) on the premises of the location of confinement and a control center (48). Voice verification, using voice analysis of speech transmitted in a telephone call from the site (58) to the center (48) is performed during periodic testing. A voice template vocabulary is established for the individual and used for voice verification. Caller line identification of each incoming call is performed to verify that call originates from the appropriate location. The confined individual is required, either randomly or at scheduled intervals, by the system to call the control center (48) and recite a statement including randomly selected words from the template vocabulary.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	FI	Finland	MN	Mongolia
AU	Australia	FR	France	MR	Mauritania
BB	Barbados	GA	Gabon	MW	Malawi
BE	Belgium	GB	United Kingdom	NL	Netherlands
BF	Burkina Faso	GN	Guinea	NO	Norway
BG	Bulgaria	GR	Greece	NZ	New Zealand
BJ	Benin	HU	Hungary	PL	Poland
BR	Brazil	IE	Ireland	PT	Portugal
CA	Canada	IT	Italy	RO	Romania
CF	Central African Republic	JP	Japan	RU	Russian Federation
CG	Congo	KP	Democratic People's Republic of Korea	SD	Sudan
CH	Switzerland	KR	Republic of Korea	SE	Sweden
CI	Côte d'Ivoire	LI	Liechtenstein	SK	Slovak Republic
CM	Cameroon	LK	Sri Lanka	SN	Senegal
CS	Czechoslovakia	LJ	Luxembourg	SU	Soviet Union
CZ	Czech Republic	MC	Monaco	TD	Chad
DE	Germany	MG	Madagascar	TG	Togo
DK	Denmark	ML	Mali	UA	Ukraine
ES	Spain			US	United States of America

METHOD AND SYSTEM FOR HOME INCARCERATIONTechnical Field

The present invention relates to remote verification of the presence of a particular individual within a predetermined confinement area, broadly described as a home incarceration system. More particularly, the invention is directed to a method and system for remotely confirming that the individual is at the prescribed premises by communicating with the individual via a telephone network, identifying the location by utilizing caller line identification and identifying the individual by voice identification speech processing.

Background Art

The concept of home incarceration has evolved as an alternative to detention in government jail and prison facilities. In cases of relatively light infractions, offenders, rather than being placed as inmates in overcrowded facilities, are confined to predetermined limited geographical areas including, for example, homes and workplaces. The burden on the prison system is relieved by enabling more space for criminals convicted of more serious crimes. Cost efficiency is also a significant factor as the expense of incarceration in such a facility is quite high. The degree of severity of punishment and the prospects of rehabilitation of the

2

light offender are more appropriate to a home incarceration environment than in a prison provided for felons.

5 In a "house arrest" situation, the detainees, of course, are more likely to interact with the community. Public security is a socially sensitive issue and it is important that the activities of captives be monitored and supervised. The whereabouts and identity of individuals should be capable of being established at 10 any time without the necessity of assignment of a law enforcement officer for constant surveillance on a one to one basis.

A prior art monitoring arrangement is shown in Fig. 1. A bracelet 20 is worn on the wrist or ankle of the detainee. A radio transmitter 22 broadcasts a coded signal which is received at a base 24. The base may be stationary or mobile. Verification of the received coded signal is performed at the base as indicated in block 26. Inasmuch as the signal has a limited range, reception of the signal at the base is indicative that the bracelet, and presumably the detainee, is within the defined area of confinement. The signal may be continuously or selectively generated.

The base is under the control of a processor through a telephone line 30. The processor may be part of a local area network including a file server 32 having data base information of all detainees in the system. At any time the system may call, via the telephone, the confinement site and ask for verification. Telephone calls may be made randomly or at scheduled intervals determined by the system. If the signal is to be continuous and the base senses an interruption in the signal, the system will initiate a call for verification.

During a call, the detainee is requested to position the bracelet appropriately near the transmitter. The transmitter then picks up the code from the bracelet and transmits it back to the base.

5 If the transmitter is beyond the range of the base, or if the code is not verified, the base can initiate a call to the system processor to indicate that the detainee is not responding or has not been verified.

10 A similar prior art arrangement is disclosed in U.S. Patent No. 4,747,120. A bracelet capable of generating a coded signal is worn by the person to be monitored. A decoder, connected with a telephone, can decode the signal when the bracelet is appropriately positioned adjacent the decoder and the decoded signal
15 can be transmitted over the telephone network to the remote system.

The above described arrangements, intended for selective or continuous personnel monitoring, have inherent disadvantages. In the prior art embodiment of
20 Fig. 1, lengthy interruptions in signal transmission can be caused by various sources of interference. As a result, the base may give frequent false indications of nonverification, requiring human intervention. Where the coded signal is transmitted by the phone line,
25 rather than by radio transmission, continuous monitoring is impractical, as an on line connection must be continuously maintained for each person monitored.

A phone call by the system to the confinement site for purposes of verification will not be productive
30 during periods in which radio transmission is interrupted by interference. As a backup for such instances, monitoring personnel may attempt to identify the voice of the called party during the telephone conversation. The listener would be required either to

4

know the confinee personally or be familiar with voice recordings of the individual to be verified. Such identification attempts likely would not be successful if the system serves a large number of detainees or if the speech of the called party is slurred by the influence of drug or alcohol abuse. Enforcement personnel frequently must be dispatched to the confinement sites to resolve the issue.

A further drawback of these systems is that the coded signal may be verified without complete assurance that the signal emanates from the location of confinement. In the case of radio transmission to the base, while the transmission range may be limited, the range may nevertheless extend beyond the bounds of confinement. In the case of telephone transmission, the system may be thwarted by placement of a decoder at a telephone, which is provided with call forward service, in an unauthorized area. A call placed by the system to the site of incarceration could be call forwarded to the unauthorized area and the code would be verified, falsely indicating that the detainee is identified and present at the appropriate location.

A further complication in these systems involves the physical structure of the bracelet. Bracelets must be constructed to resist tampering. The device must be affixed to the particular individual so that the identity of that person can be assured when receiving the signal transmission. The device is cumbersome in order to prevent easy removal. In addition, each bracelet must have a self-contained power supply sufficient for operation over an extended time period.

U.S. Patent No. 4,843,377 contemplates the use of a voiceprint as a means for remote identification of a prisoner. Audio spectral analysis is performed and

5

applied to speech transmitted over a telephone line to determine a match with a probationer's voiceprint. Several commercially available systems are discussed.

5 While voice analysis may be a reliable means to determine the identity of an individual, such a system, in itself, cannot verify that the individual is at the prescribed location. Call forwarding, in the network or on the premises, can result in the appearance of a party being in the prescribed location, while in fact, being
10 elsewhere.

Disclosure of the Invention

Accordingly, an object of the invention is to provide a home incarceration system having the capability of remotely verifying the identity of an
15 individual and the location of the individual at any time.

Another object of the invention is to provide a home incarceration system that is not subject to false indications of nonverification which may result from
20 outside interference.

A further object of the invention is to enable remote monitoring of detainees in confinement without the necessity of a device that may be subject to physical tampering or breakdown.

25 Yet another object of the invention is to enable simultaneous, remote and automatic monitoring of a large number of confinees while requiring a minimum number of monitoring personnel. A related object is to provide an automatic warning or message to remote personnel in the event of a system determination that a confinee is
30 absent from the required location.

A further object is to permit multiple legitimate sites of incarceration based, for example, on time of

6

day or week and which can be easily verified by known telephone number.

An additional object of the invention is to provide a plurality of local control centers, each serving a home incarceration monitoring and control function for a prescribed geographical area and having the capability to selectively transfer the functions of any particular control center to another local center or a master network center for prescribed time periods.

The above and other objects of the invention are satisfied in part by providing a telephone communication network linking each confinement location to a remote home incarceration center. The system includes a controller and storage at the control center or at a remote location linked thereto. The system maintains a database of inmates currently included in the program. The prisoner database includes each inmate's name, telephone number of the site of incarceration, and date and period of incarceration. In the event that the inmate is permitted a work schedule, the database includes the telephone numbers of each permitted location and corresponding scheduled time periods.

When a new inmate is added to the system, the inmate is escorted to the incarceration location by civil authorities. Once there, telephone communication is established with the home incarceration center to establish an identity for the inmate. Voice training is undertaken to establish voice templates for the individual inmate. A variety of words are selected to form a test vocabulary. The words are recited by the inmate from the incarceration site and transmitted to the incarceration center where a voice template for each word is created and stored. This procedure avoids any detrimental influence resulting from variations in

telephone transmission characteristics from different origins.

Once in the system database, and with the voice templates established, the inmate is subjected to periodic testing. Testing may be performed at predetermined schedules and at random intervals. A test is initiated by retrieving the inmate's number from the database and calling the incarceration site. An announcement is then made, requesting the inmate to call back in to the home incarceration center within a fixed time period to conduct the voice identification test. The system will be prepared to accept the incoming call. Caller line identification at the control center determines if the return call is made from the incarceration site. During the call, the inmate is required to recite a statement, prepared at the incarceration control center, including randomly chosen words from the template vocabulary. Comparison is made, using speech analysis, between the recited statement and the stored templates. As the statement is unknown to the inmate in advance, an attempt to use a voice recording as a response, with the inmate absent, would be futile.

The testing can be controlled manually at the incarceration center or be handled completely automatically. In an automated test procedure, the system would send notification, visibly or audibly, to an administrator of any test failures. Such notification may be transmitted through the network to the administrator at a location remote from the test center. A log file is maintained by the system for the purpose of recording all activity by the system, whether manually or automatically instituted.

8

The system can operate in the environment on one or more sites of the law enforcement authority premises on a dedicated line basis. Alternatively, a single system can be shared on a network basis by several law enforcement agencies by appropriate partitioning. An additional aspect of the invention is call forwarding control by one center to another for "after hours" monitoring or for other purposes.

Additional objects and advantages of the present invention will become readily apparent to those skilled in this art from the following detailed description, wherein only the preferred embodiment of the invention is shown and described, simply by way of illustration of the best mode contemplated for carrying out the invention. As will be realized, the invention is capable of other and different embodiments, and its several details are capable of modifications in various obvious respects, all without departing from the invention. Accordingly, the drawings and description are to be regarded as illustrative in nature, and not as restrictive.

Brief Description of Drawings

Fig. 1 is a block diagram illustrating a prior art monitoring system.

Fig. 2 is a block diagram of a system according to the present invention, depicting a network including a control office and controller within the network.

Fig. 3 is a block diagram according to the present invention, representing broadly the components of the system including the control functions.

Fig. 4 is a function map of the home incarceration architecture of the present invention.

9

Fig. 5 is a function map of the speaker verification process of the present invention.

Fig. 6 is a network layout of the present invention, illustrating transfer capability among various geographically separated control centers.

Best Mode for Carrying out the Invention

Fig. 2 broadly illustrates a home incarceration system including incarceration control center 48. An adjunct processor/server is shown at 50 interconnected through a network 52 to basic incarceration controllers 54, only one of which is represented in the figure for simplicity of illustration. Network 52 may be a local area network, such as ethernet, or a wide area network, such as a private line T1 network. The basic incarceration controller is shown connected, via telephone facilities (e.g., lines), to a central office 56, which is also connected via telephone lines to a particular local incarceration site 58. In addition to serving the area of central office 56, control center 48 may be extended to include additional central office areas such as the central office 60, shown connected therewith through interoffice facility line 62.

The basic incarceration controller includes telephone interfaces, a processor and voice processing/response capabilities, using appropriate hardware and software. The basic incarceration controller may also include storage for the inmate database and speech templates to perform all control functions as an independent unit. The system capacity can be extended beyond the limits of the basic incarceration controller by the adjunct processor/server, which includes memory storage and program control for the basic incarceration controller.

Fig. 3 is a further development of the control elements of the system shown in connection with central office 56. The central office is connected to subscriber loop connector 70 and modem 74 through bridge circuit 72. Bridge circuit 72 allows an incoming call to be split, permitting the incoming call signals to be applied to the subscriber loop connector 70 and modem 74. The subscriber loop connector is connected to a communications port, not shown, in controller 78 through amplifier/filter circuit 80. The modem 74 is connected to an originating call identification device 76.

The subscriber loop connector is a well known unit that performs both incoming and outgoing call functions. This unit serves to control the telephone connection with the central office and user telephone line. For incoming calls, for example, the unit detects ringing, on-hook and off-hook. The unit, under direction of the controller, performs outgoing call functions. These functions, in an alternative embodiment, can be incorporated into an appropriate function board in the system microprocessor.

In operation, for incoming calls block 76 identifies the originating telephone number from information transmitted between the first and second ringing signals. A detailed description of the preferred composition of this device is contained in U.S. application serial number 07/515,027, filed April 26, 1990 which application is herein incorporated by reference. Alternatively, the functions of block 76 can be incorporated in the control program.

The subscriber loop connector establishes off-hook connection between the central office and the controller after the second ring. At this time the calling line has been identified by block 76 and this information is

11

transmitted to the controller 78. The controller includes a processor for comparing the caller identification information with the stored database information associated with the inmates.

5 The processor also performs speech analysis, comparing the transmitted voice of the caller with stored voice templates. Speech and voice processing may be performed in accordance with technology well known in the art. Examples of suitable speech verification
10 algorithms may be found in "Digital Processing of Speech Signals," by Rabiner and Schafer, Bell Laboratories, Prentice-Hall, Inc. 1978, particularly at page 457. Further description of voiceprint analysis for voice
15 identification may be found in U.S. patent number 3,525,811.

 The filters and amplifiers forming block 80 condition the transmitted audio signals to limit the band width and strengthen the signals appropriately to the requirements of the processing to be performed in
20 controller 78. Two way voice communication is transmitted between the incarceration site and the controller through the path including the central office, bridge circuit, subscriber loop connector, and
amplifier/filter circuit.

25 Upon entering a new inmate into the system, a telephone call is established whereby "voice training" is performed. Voice templates of a selected word vocabulary are created and stored in the data base. The data base also includes the inmate's telephone number at
30 home, work or other permitted location, scheduled hours at each permitted location, telephone number of probation officer or other official to notified in case of a violation, and any other pertinent information.

12

The data base can be updated at any time without interrupting the calling activity of the system.

Testing is performed by calling the telephone at which the inmate is scheduled to be and requiring the inmate to call back to the control center. In the return call, the caller line is identified by block 76 and the inmate is required to repeat a statement including selected words from the template vocabulary. Verification of the caller's voice is made by comparison therewith with the stored templates, using voice analysis techniques described above. Dynamic adaptive updating of the templates may be periodically performed upon successful voice verification.

Calls may be placed by the control center on the basis of a predetermined schedule as well as randomly. The system has flexibility to determine frequency of random calls made per day and to change the frequency for each inmate as deemed appropriate. For example, inmates who have violated curfew might be assigned a higher frequency of random calls. In addition, inmates may be required to call in regularly at predetermined times.

Violations are reported automatically to administrative personnel by transmission of a message to a remote printer or terminal. Notification may also be effected by audible alarm, paging or delivery of a voice mail message. All activity is recorded in a log file maintained in the system.

The home incarceration monitoring scheme may include a continuous signalling device worn by the inmate as part of a hybrid system. This added redundancy would make the home incarceration concept more socially acceptable as well as afford continuous surveillance. During times in which broadcast

transmission of the continuous signal is interrupted by interference or other instances in which no signal is received, the system can initiate a call to the inmate for verification by calling line identification and voice analysis.

Figs. 4 and 5 are charts illustrating the functions of the system including initialization, database administration, training and testing. These functions are under the control of a main program executed by the system processor. Database administration includes adding and deleting information, as well as a print capability. In Fig. 5, speech processing includes voice training to create templates and testing, using the templates and transmitted speech.

Calling party number identification may be obtained through ISDN or analog lines equipped with caller line identification or similar services. Number identification can also be transmitted using out of band signaling, packet switching or Simplified Message Service Interface (SMSI), ISDN primary rate access and bulk calling line identification. In some cases a trunk arrangement may be used in a PBX environment.

Fig. 6 illustrates an intelligent network application of the home incarceration service. Central offices 56, each serving end user incarceration sites 58, are shown interconnected with each other. A local control center 54 may be customer premises equipment or network based and can perform voice verification and caller party number identification. Similarly, a larger area control center, which may be customer premises equipment or network based, is shown at 55.

Sufficient hardware and software to serve the entire system is provided at network base processor 50, which may be used in conjunction with signal control

14

point 80. The signal control point is attached to the network through signal transfer point 82 to monitor all signaling within the network and to intelligently control the action to taken based on the signal.

- 5 Additional signal transfer points may be included to accommodate network size.

The signal transfer point is connected to each of the central offices through SS7 or other data links for database information transfer. The local control center
10 may be operational for limited hours. Transfer of the functions of this center for after hours coverage can be made to the area control center under control of the network base processor via the signal transfer point or by call forwarding from the local office.

- 15 Only the preferred embodiment of the invention and but a few examples of its versatility are shown and described in the present disclosure. It is to be understood that the invention is capable of use in various other combinations and environments and is
20 capable of changes or modifications within the scope of the inventive concept as expressed herein.

PCT/US92/07645

15

WHAT IS CLAIMED IS:

1. A method for remotely verifying, at a verification site, the attendance of a particular person at a predetermined area, said area being provided with a telephone, comprising the steps of:

- 5 establishing an on line telephone connection between said verification site and said telephone;
 determining a voice characteristic of said person at said verification site in response to speech transmission through said telephone connection; and
10 testing for the presence of said person at said area, said step of testing comprising the steps of:
 identifying a calling telephone line directory number in response to an incoming telephone call;
15 establishing an on line connection for said incoming telephone call; and
 analyzing a voice transmitted during said incoming call.

2. A method as recited in claim 1, wherein said step of testing further comprises a step of determining whether the identified telephone line directory number corresponds to a predetermined line directory number associated with said telephone.

3. A method as recited in claim 2, wherein said step of testing is selectively performed at random times and further comprises:

- 5 calling said telephone from said verification site;
 establishing a further on line connection between said verification site and said telephone;

16

- requiring a return call from said telephone to said verification site within a set time period after termination of said further on line connection; and
- 10 determining whether a return call has been established within said set time period.

4. A method as recited in claim 3, further comprising the step of generating a warning indication upon determination that a return call has not been established within said set time period.

5. A method as recited in claim 2, wherein said step of determining a voice characteristic comprises:
- selecting a plurality of words to form a voice vocabulary;
- 5 creating a voice template for each word of said vocabulary as spoken by said person in said speech transmission; and
- storing the templates created.

6. A method as recited in claim 5, wherein said step of testing further comprises:
- preparing a statement containing one or more words included in said vocabulary; and
- 5 requiring the caller of said incoming call to recite said statement;
- and said step of analyzing comprises comparing the recited statement with the stored templates.

7. A method as recited in claim 6, wherein said step of determining a voice characteristic further comprises dynamically updating the stored templates.

T03T00"X000000

17

5 8. A method as recited in claim 6, further including the step of generating a warning indication upon a condition that either said identified telephone line does not correspond to a line associated with said telephone or that the voice transmitted during said incoming call does not match said stored templates.

9. A method as recited in claim 8, wherein said generating step comprises displaying a message on a terminal.

10. A method as recited in claim 9, wherein said terminal is remote from said verification site.

11. A method as recited in claim 8, wherein said generating step comprises printing out a message.

12. A method as recited in claim 8, wherein said generating step comprises transmitting a warning message.

13. A method as recited in claim 12, wherein said message is a paging communication.

14. A method as recited in claim 12, wherein said message is a voice mail message.

15. A method as recited in claim 12, wherein said warning message comprises an audible alarm.

16. A method as recited in claim 12, wherein said step of transmitting a warning message comprises automatically generating a radio dispatch to a patrol vehicle.

18

17. A method as recited in claim 6, further comprising storing results of the tests performed.

18. A system for monitoring at one or more remote locations the presence or absence of a particular person within a defined area comprising:

a telephone at said defined area;

5 verification means remote from said area for
verifying the identity of an individual at said area;
and

a communications network for establishing communication between said telephone and said verification means;

said verification means comprising:

voice processing means for analyzing an incoming voice transmission from said communications network; and

15 caller line directory number
identification means for identifying an
incoming caller telephone line.

19. A system as recited in claim 18, wherein said voice processing means comprises:

means for creating voice templates of a preselected word vocabulary for said person;

5 means for storing said voice templates; and

means for comparing spoken words of said voice transmission with said voice templates.

20. A system as recited in claim 18, wherein said verification means further comprises storage means for storing information including identification of said telephone line directory number as a reference for

5

5

22. A system as recited in claim 21, wherein said means for generating includes a display terminal.

23. A system as recited in claim 21, wherein said means for generating includes a printer.

24. A system as recited in claim 20, including two or more telephones at geographically separated locations within said defined area, said storing means including stored identification of each of said telephones.

5

26. A system as recited in claim 25, wherein said communications network comprises a plurality of signal transfer points and call forward means for transferring

20

verification operation from one of said verification
5 means to another of said verification means through a
signal transfer point in said network.

27. A system as recited in claim 18, further including means affixed to said person for transmitting a continuous signal and means for monitoring said continuous signal.

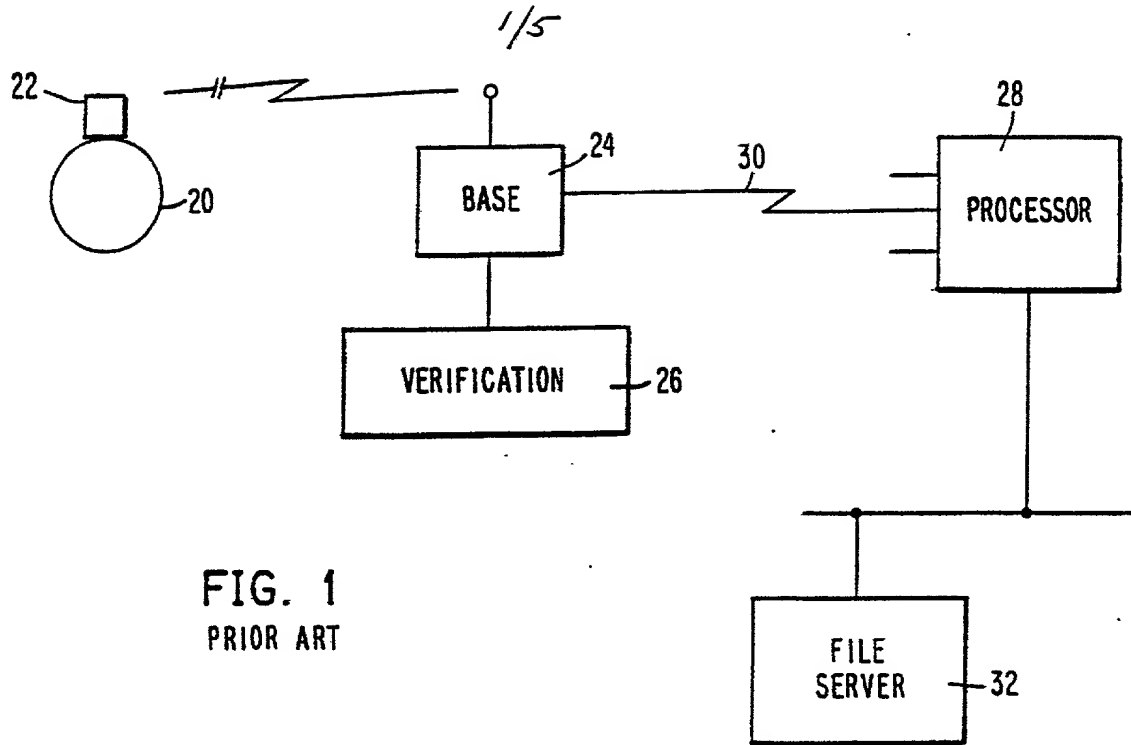


FIG. 1
PRIOR ART

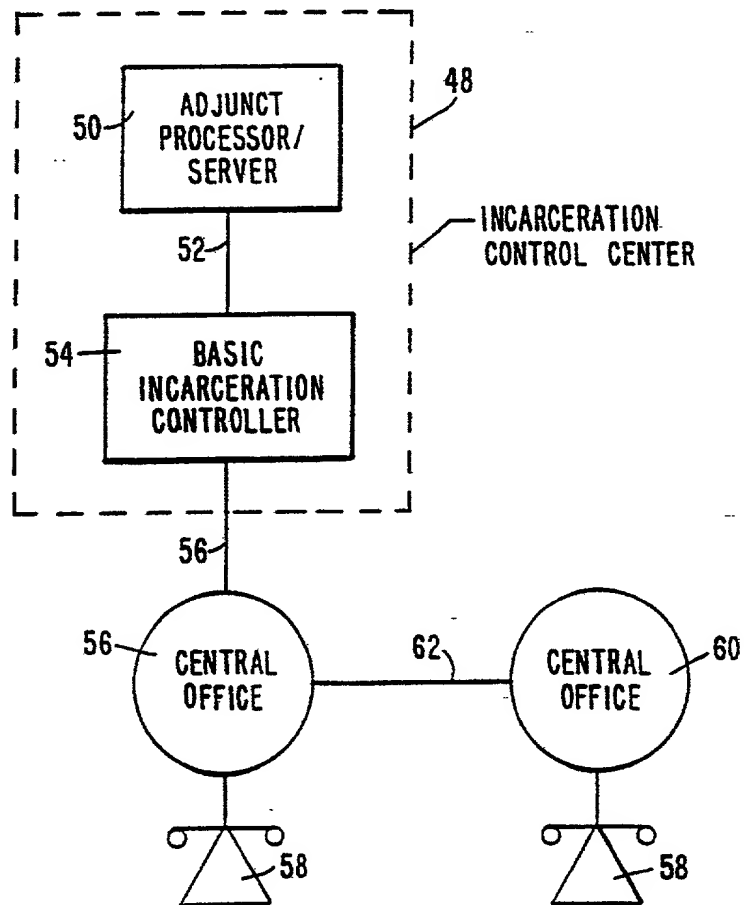


FIG. 2

2/5

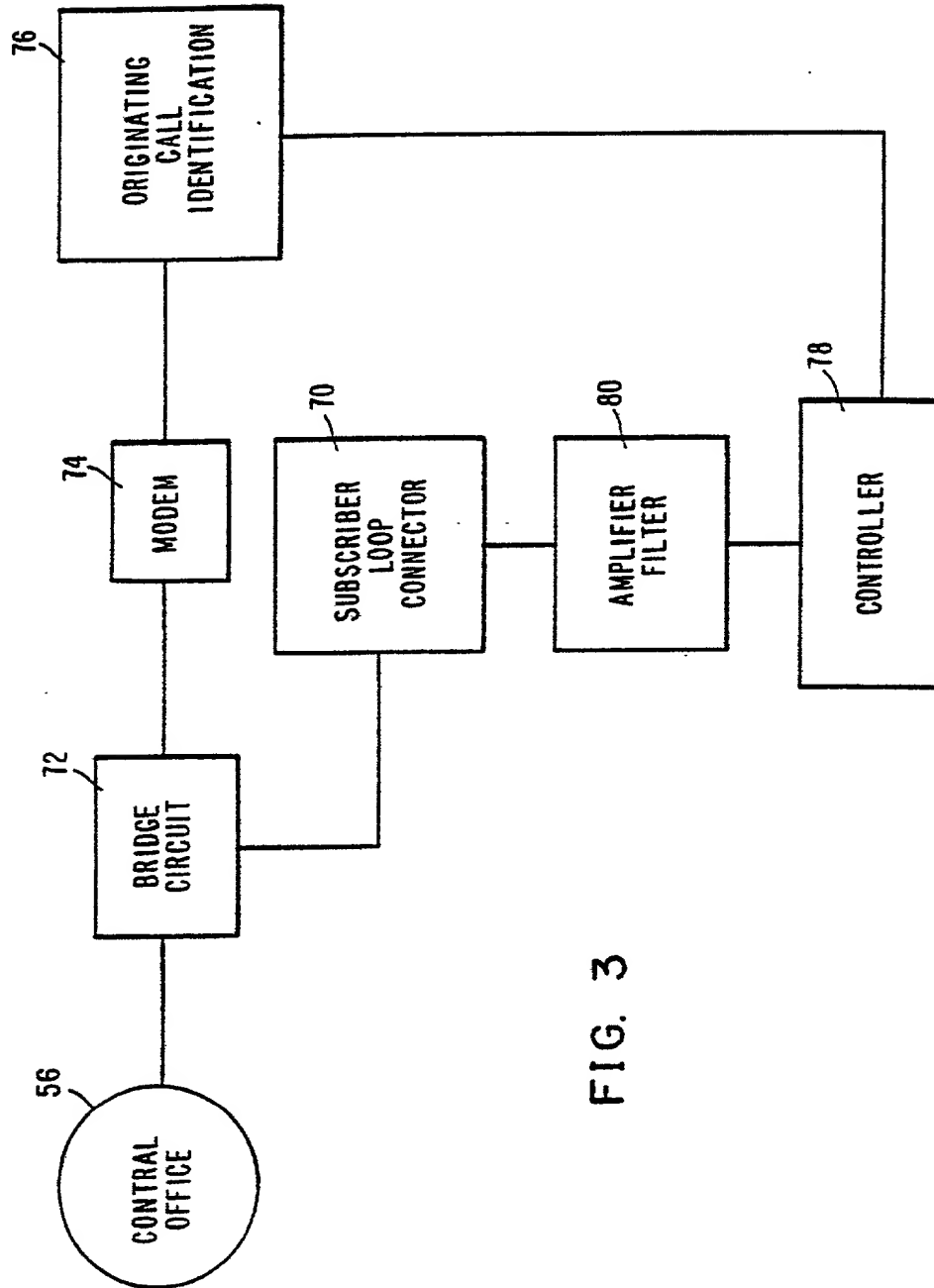


FIG. 3

3/5

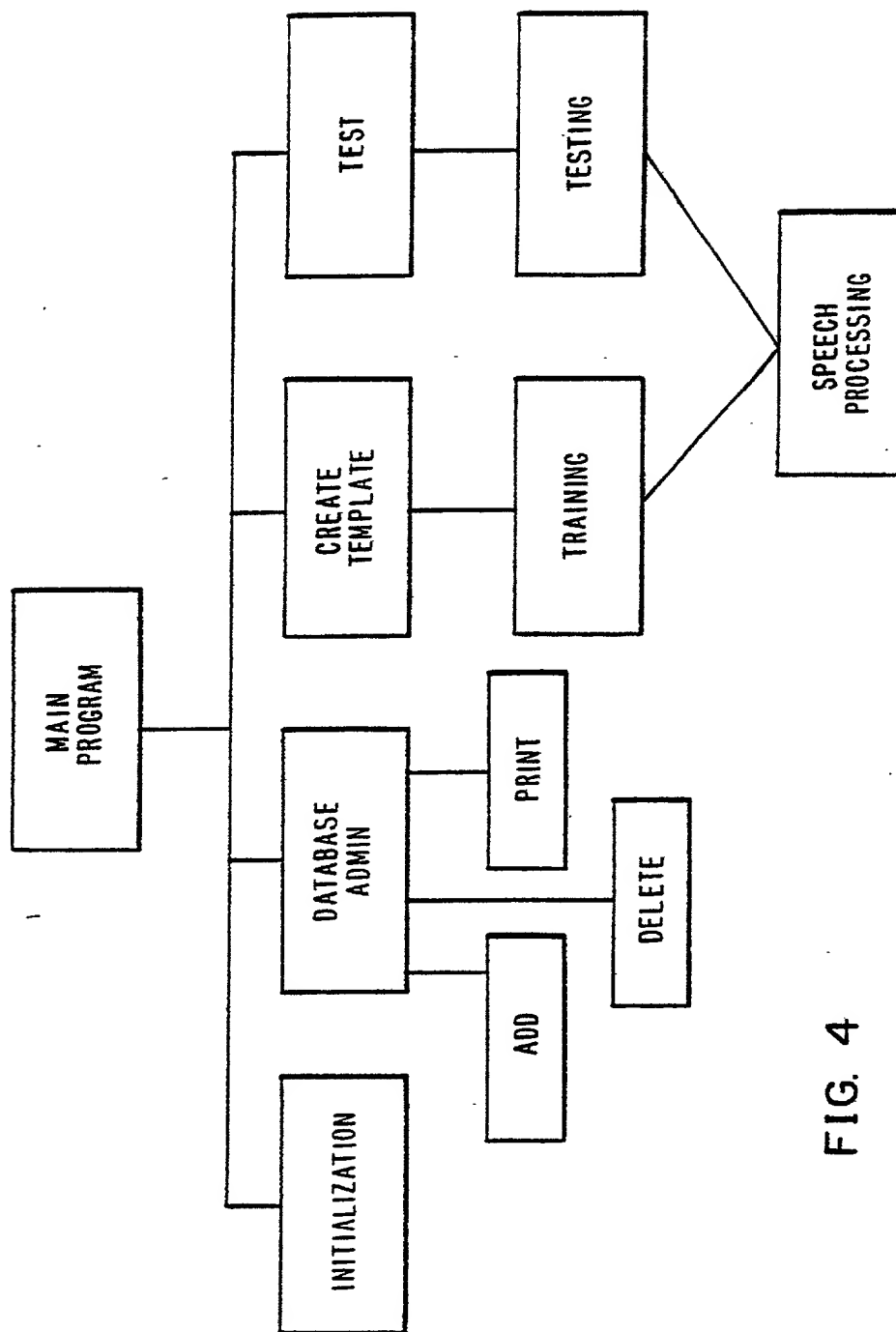


FIG. 4

4/5

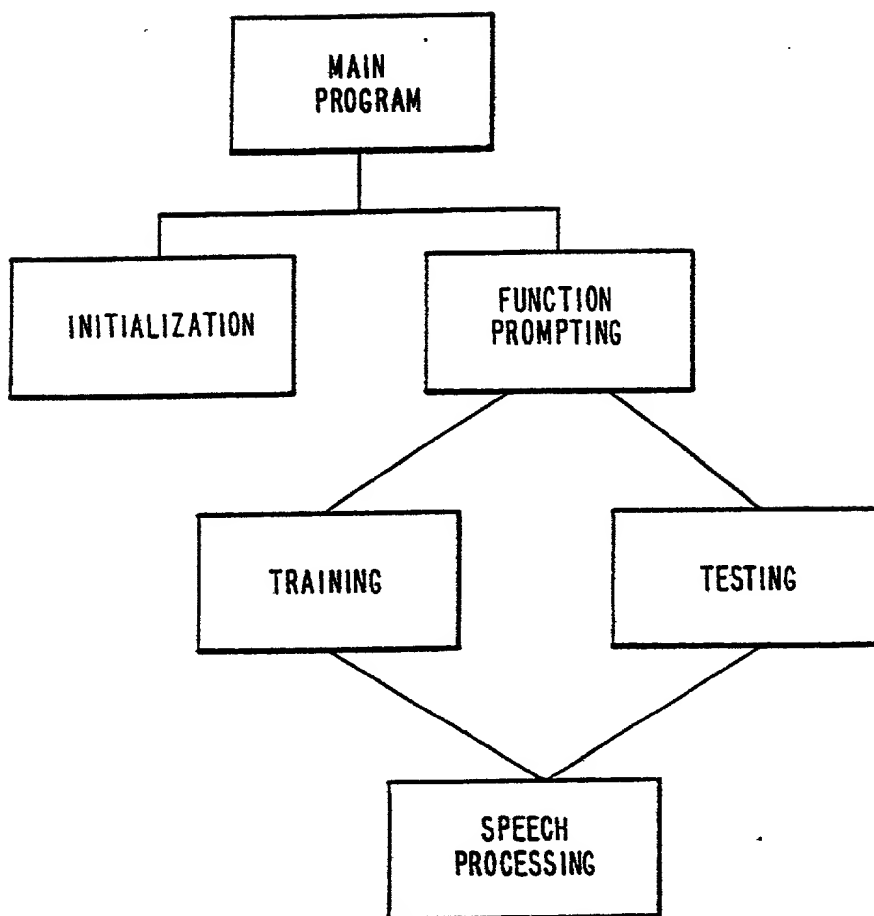


FIG. 5

5/5

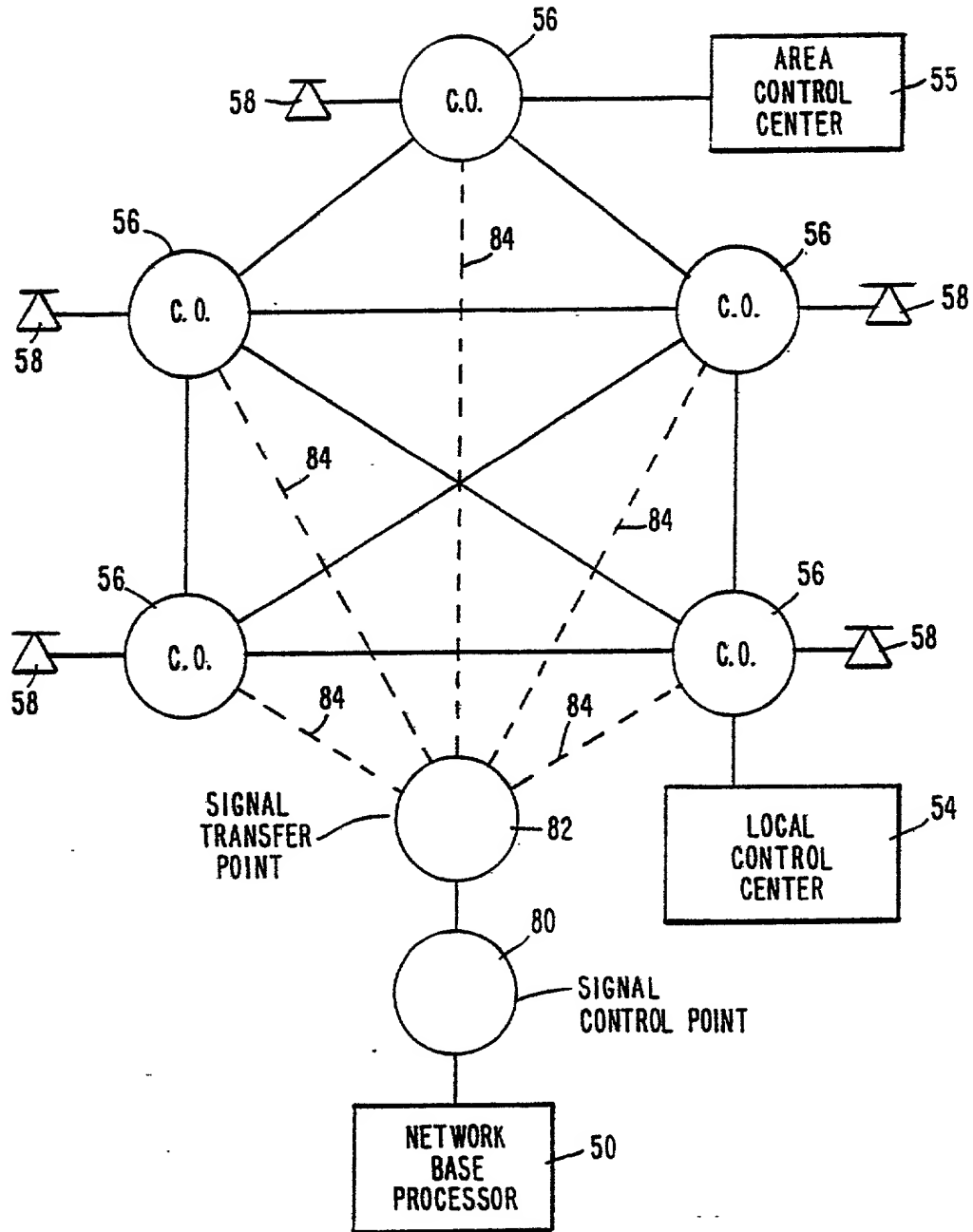


FIG. 6

INTERNATIONAL SEARCH REPORT

PCT/US92/07645

A. CLASSIFICATION OF SUBJECT MATTER

IPC(5) :H04M 11/04

US CL :379/38

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 379/42,49,50,106,142;340/505,539,573,592,825.08,825.34,825.36;381/42;381/2-4

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

none

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US,A, 5,023,901 (SLOAN ET AL) 11 JUNE 1991 See entire document.	1-27
A,P	US,A, 5,054,055 (HANLE ET AL) 01 OCTOBER 1991 See entire document.	1-27

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:	T	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
A document defining the general state of the art which is not considered to be part of particular relevance	X*	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
E earlier document published on or after the international filing date	Y*	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
L document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	Z*	document member of the same patent family
O document referring to an oral disclosure, use, exhibition or other means		
P document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search

18 NOVEMBER 1992

Date of mailing of the international search report

4 DEC 1992

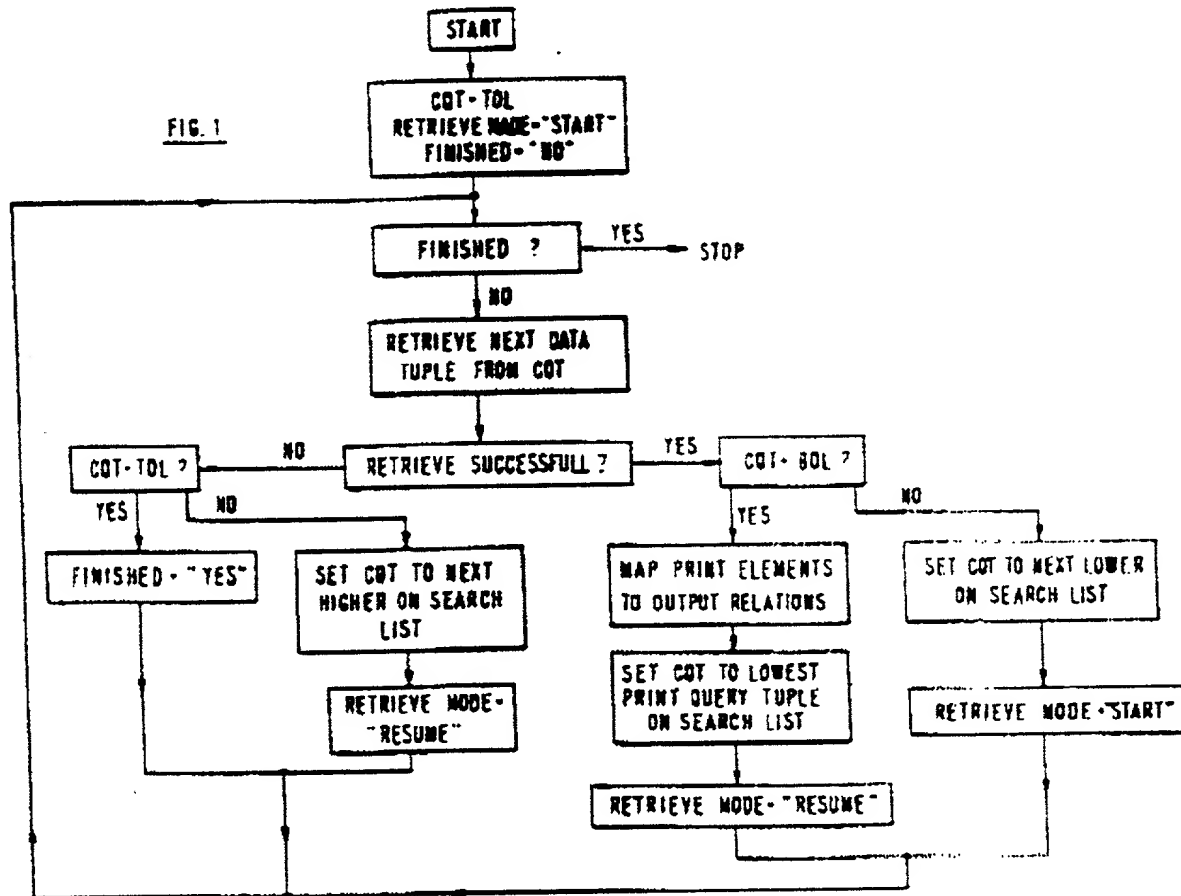
Name and mailing address of the ISA/
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231Authorized officer *W. F. Chan*
WING F. CHAN

Facsimile No. NOT APPLICABLE

Telephone No. (703) 305-4732

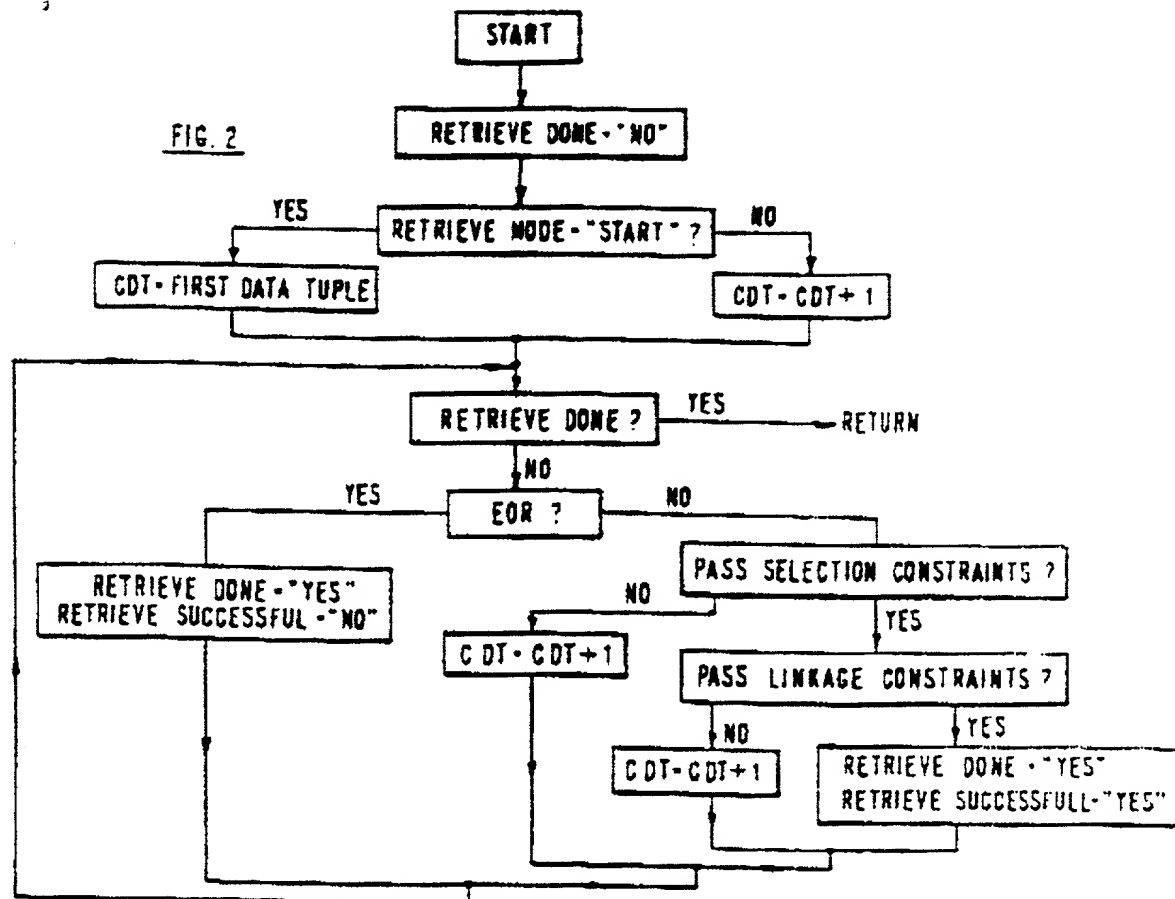
N-ARY JOIN FOR PROCESSING QUERY BY EXAMPLE

K. E. Niebuhr and S. E. Smith



Given a relational data base, the present method provides an efficient way of performing the join operations needed to solve a query-by-example query. In an earlier publication [1], the joining was accomplished by a sequence of binary joins, each of which produced a temporary relation which could be involved in subsequent joins. That approach had the following potential disadvantages:

- (1) Overhead associated with storing the data tuples in the temporary relations resulting from the joins.
- (2) Because all possible solutions were carried throughout the sequence of joins, the cardinalities of the temporary relations could become very large for a modest size data base.



- (3) Inversions were not available in the temporary results so they had to be created, if needed. Moreover, existing inversions were frequently not used.

The proposed method overcomes all of these shortcomings with a single operation which is called the *n*-ary join.

Terminology

A. Query Tuples and Data Tuples

It is necessary to distinguish between two kinds of tuples. The user constructs a query-by-example query in the form of "query tuples" within skeleton tables [1,2]. Corresponding to each skeleton table and its query tuples is a relation containing "data tuples". The query tuples specify the constraints which the data tuples must satisfy in order to be included in the answer to a query.

B. Selection and Linkage Constraints

The data tuples which comprise the answer to a query must satisfy

two kinds of constraints, "selection" constraints and "linkage" constraints. Selection constraints are intra-tuple conditions specified by a query tuple. Examples of selection constraints are:

- *Column 3 must have a value equal to 10.
- *The value in column 5 must be greater than the value in column 2.

In general, a selection constraint is a function over the entries in a single data tuple which evaluates to true or false.

Linkage constraints are inter-tuple conditions specified between multiple query tuples. Examples of linkage constraints are:

- *The value in column 3 of the data tuple in R2 must be greater than the value of column 5 of the data tuple currently selected from R1.
- *The set of column 4 values for all data tuples in a relation, R2, which have the same value in column 3 of the data tuple in R2 under investigation (column 3 is a "grouper" column) must be contained in a similarly defined set for the currently selected data tuple of a relation R1. ("set join", see [1]).

C. Search List

Prior to initiating the n-ary join operation, the query tuples are organized in a list called the Search List. This list determines the order in which the query tuples and their underlying data relations are processed with the query tuple at the top being processed first. Explicit join query tuples are not placed on the Search List. If explicit join tuples exist in a query, their "print" elements are temporarily transferred to the appropriate normal query tuples.

N-ary Join

The n-ary join will be described using the flow charts of Figs. 1 and 2.

Fig. 1 shows the top level action involved in selecting the current query tuple for which a solution-satisfying data tuple is to be retrieved from its underlying relation. In this figure:

- *CQT is the current query tuple which is that tuple on the Search List which is being processed during a given pass through the flow chart.
- *TOL is the query tuple at the top of the Search List.
- *BOL is the query tuple at the bottom of the Search List.
- *The Retrieve Mode setting dictates whether the next data tuple retrieval search should be started at the beginning of the relation ('Start') or continued from the last retrieval made from subject relation ('Resume').
- *Retrieve Next Data Tuple From CQT returns with a Retrieve Suc-

cessful='Yes' when the first data tuple, which satisfies the selection and linkage conditions, is found or Retrieve Successful-'No' if the end-of-relation was reached.

*Whenever a satisfactory data tuple is found for the BOL query tuple, the print elements in the current data tuples are mapped into output relations and the retrieval resumes scanning with that "print" query tuple which is lowest on the Search List.

Fig. 2 shows the action associated with the data tuple retrieval on the relation which underlies the current query tuple. In this figure:

- *CDT is the data tuple pointed to currently.
- *If the Retrieve Mode='Start' then CDT is initially set to the top of the relation. (If Inversions had existed in a direct access system, an equality restriction, for example, could be associated with a list of valid data tuple IDs and CDT would be set to the top of that list.)
- *If Retrieve Mode='Resume' then CDT is incremented from its current data tuple to the next data tuple in the relation or the next data tuple available in the data tuple ID list (if such a list has been created). Moreover, the CDT for all query tuples must be stored on a continuing basis.
- *If the attempt to acquire the next data tuple results in an end-of-relation (EOR='Yes'), this procedure returns Retrieve Successful-'No'.
- *Candidate data tuples are first tested to determine if they pass Selection constraints. They are then tested for inter-tuple dependencies with the previously selected data tuples (one per query tuple) from the relations that underlie those query tuples which are above the current query tuple on the Search List. Certain types of linkage tests can be eliminated. For example, if a query tuple has the same linkage test involving two or more query tuples (which are higher on the Search List), only one of those linkage tests is necessary.)
- *If the candidate data tuple fails either the selection or linkage tests, CDT is incremented to the next available data tuple.

Ordering of Search List

The order of the query tuples in the Search List has a significant effect on the amount of processing required to find a solution to a query. Three criteria are used to determine the order of the Search List.

- (1) Cardinality. Query tuples associated with the smallest cardinality relations are placed at the top of the list and those associated with large relations are placed at the bottom.
- (2) Print elements. An attempt is made to order the list so that the lowest query tuple with a print element is as high as possible in the list. This is because when a data tuple for the lowest print query tuple is processed, the processing of lower query tuples can

be terminated after the first set of data tuples satisfying all selection and linkage constraints has been found.

- (3) Linkage. An attempt is made to order the list such that the tuples at the top of the list have the maximum number of inter-tuple linkage constraints. This will tend to reveal inadmissible data tuples as soon as possible, and avoid processing of lower level query tuples and their associated relations.

For any given query the orderings determined by each of these criteria will generally be different. It is thus necessary to arrive at a compromise ordering which weights the importance of these three criteria. We use the following algorithm to arrive at an ordering according to these three criteria.

- (1) Each query tuple is assigned a number. If it does not have print elements, this number is its cardinality. If it has print elements, the number is its cardinality divided by 4. (Four is a "print factor".)
- (2) The query tuple with the smallest number is assigned to the top of the Search List.
- (3) The next position on the Search List is assigned to the query tuple which has the maximum number of linkage constraints with tuples already on the list. In case of a tie, the tuple with the smallest cardinality number (from step 1) is selected.
- (4) Repeat step 3 until all query tuples are assigned to the Search List.

Optional Selection Preprocessing

For query tuples other than the one at the top of the Search List, it will sometimes be beneficial to perform a preliminary scan of the underlying relation (using inversions when possible) to produce a smaller (in degree and cardinality) temporary relation containing only the data tuples which satisfy the intra-tuple constraints. The scans which would occur many times over each such relation will then be over the smaller temporary relation rather than the larger base relation. This will be helpful when the existing inversions on the base relations can not be used to process inter-tuple constraints.

- [1] K. E. Niebuhr, K. W. Scholz and S. E. Smith, "Algorithm for Processing Query By Example", IBM Technical Disclosure Bulletin, July 1976, Vol. 19, No. 2, pp. 736-741.
- [2] M. M. Zloof, "Query By Example", IBM Research Report RC 4917, July 1974.

United States Patent & Trademark Office
Office of Initial Patent Examination

Application papers not suitable for publication

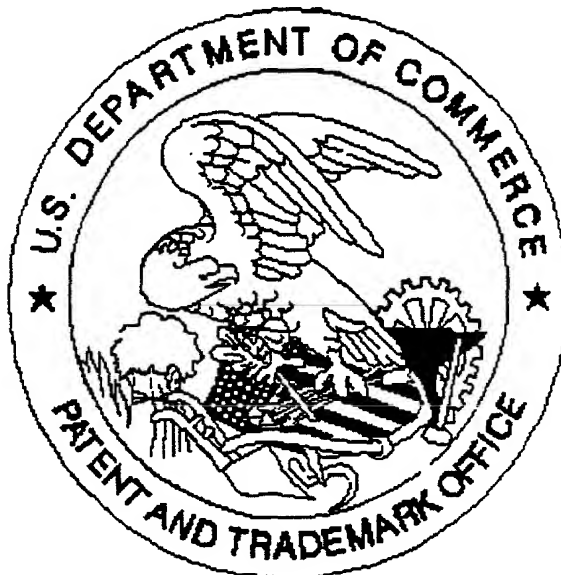
SN 09930395

Mail Date 08/20/02

- ☐ Non-English Specification
- ☐ Specification contains drawing(s) on page(s) _____ or table(s) _____
- ☐ Landscape orientation of text ☐ Specification ☐ Claims ☐ Abstract
- ☐ Handwritten ☐ Specification ☐ Claims ☐ Abstract
- ☐ More than one column ☐ Specification ☐ Claims ☐ Abstract
- ☐ Improper line spacing ☐ Specification ☐ Claims ☐ Abstract
- ☒ Claims not on separate page(s)
- ☐ Abstract not on separate page(s)
- ☐ Improper paper size -- Must be either A4 (21 cm x 29.7 cm) or 8-1/2"x 11"
- ☐ Specification page(s) _____ ☐ Abstract
- ☐ Drawing page(s) _____ ☐ Claim(s)
- ☐ Improper margins
- ☐ Specification page(s) _____ ☐ Abstract
- ☐ Drawing page(s) _____ ☐ Claim(s)
- ☐ Not reproducible Section
- Reason ☐ Specification page(s) _____
- ☐ Paper too thin ☐ Drawing page(s) _____
- ☐ Glossy pages ☐ Abstract
- ☐ Non-white background ☐ Claim(s)
- ☐ Drawing objection(s)
- ☐ Missing lead lines, drawing(s) _____
- ☐ Line quality is too light, drawing(s) _____
- ☐ More than 1 drawing and not numbered correctly
- ☐ Non-English text, drawing(s) _____
- ☐ Excessive text, drawing(s) _____
- ☐ Photographs capable of illustration, drawing(s) _____

09930395

United States Patent & Trademark Office
Office of Initial Patent Examination -- Scanning Division



Application deficiencies found during scanning:

☐ Page(s) _____ of _____ were not present
for scanning. (Document title)

☐ Page(s) _____ of _____ were not present
for scanning. (Document title)

☒ Scanned copy is best available.

Miscellaneous.